Deploying Red Hat 3Scale API Management on Red Hat Openshift with Service Mesh

Lab Guide

October 2020

Deploying Red Hat 3Scale API Management on Red Hat Openshift with Service Mesh

PUBLISHED BY:

Stone Door Group - letsdothis@stonedoorgroup.com

Copyright 2020 Stone Door Group, LLC. All rights reserved.

No parts of the contents of this book may be reproduced or transmitted in any form or by any means without the written permission from Stone Door Group.

Deploying Red Hat 3Scale API Management on Red Hat Openshift with Service Mesh

Overview 4

1.0 Deploying 3Scale on Red Hat OpenShift w/Service Mesh 4

1.1 3Scale Installation Instructions 5

1.2 Conclusion 14

1.3 Interfacing 3Scale with Service Mesh on OpenShift 15

2.0 Install Sample Application Application on OpenShift 15

2.1 WeNote Installation on OpenShift 16

2.2 Conclusion 21

3.0 Create 3Scale Corporate Site and Setup API Gateway 21

3.1 Create A Development Group on 3Scale 21

3.2 Create a New Product 23

3.3 Promoting to Staging, Production and Testing the WeNote API 32

4.0 Conclusion 35

Appendix A - Example External Application Portal 36

Appendix B - Additional Documentation/Information 40

Deploying Red Hat 3Scale API Management on Red Hat Openshift with Service Mesh

Overview

The purpose of this lab guide is to demonstrate the steps to install a new application running on Red Hat 3Scale. These lab steps work in conjunction with the IBM Cloud based lab setup provided by Stone Door Group for its live developer workshop. In order to successfully complete these steps, the student must have access to the Stone Door Group lab environment.

1.0 Deploying 3Scale on Red Hat OpenShift w/Service Mesh

In this section, we will install 3Scale. 3Scale runs as a project within Red Hat OpenShift and can be deployed via the OpenShift Operator framework. OpenShift operators enable application installation within minutes and via a few mouse clicks.

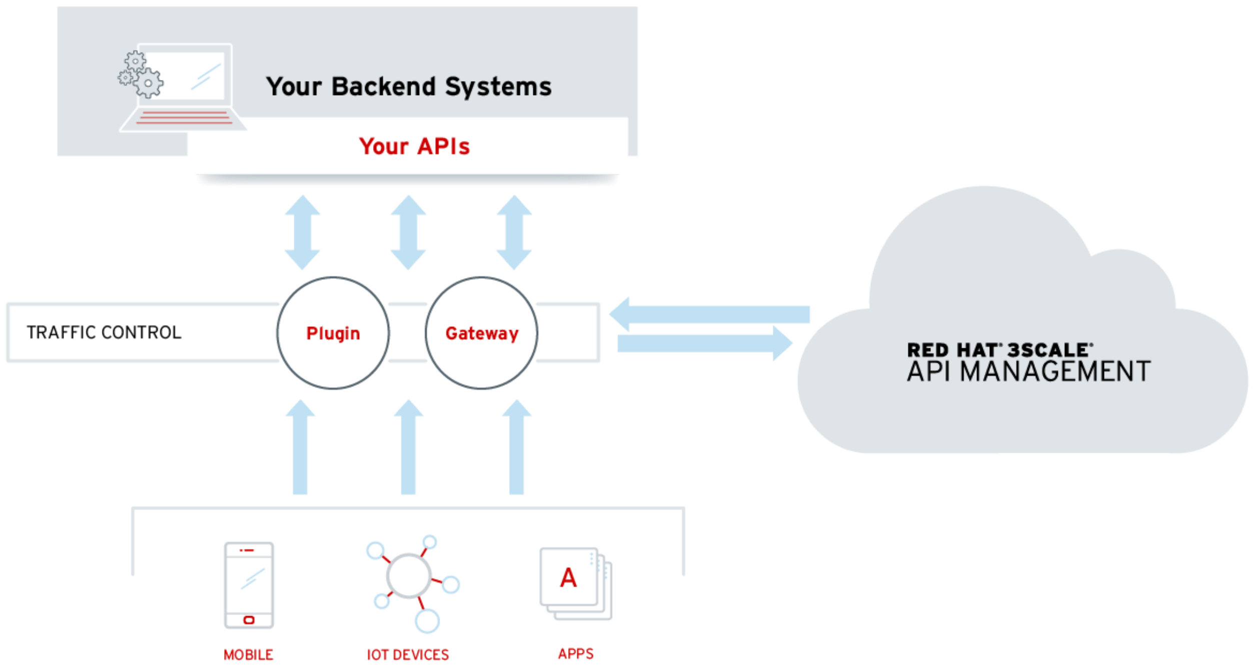
There are two main software components to 3Scale:

- **3Scale -** this is the core 3Scale application that enables the API publishing functions
- **APIcast -** this provides the capacity to provide external API endpoints, metering, and security

In addition to 3Scale, we will configure ServiceMesh for application observability. OpenShift ServiceMesh is based on the Istio project and includes Prometheus, Kiali, and Grafana.

NOTE - This tutorial assumes you have an OpenShift cluster running. If you do not have OpenShift 4.x running, here for a brief tutorial on how to install and set up an OCP cluster on IBM cloud.

The following diagram describes the target 3Scale architecture:

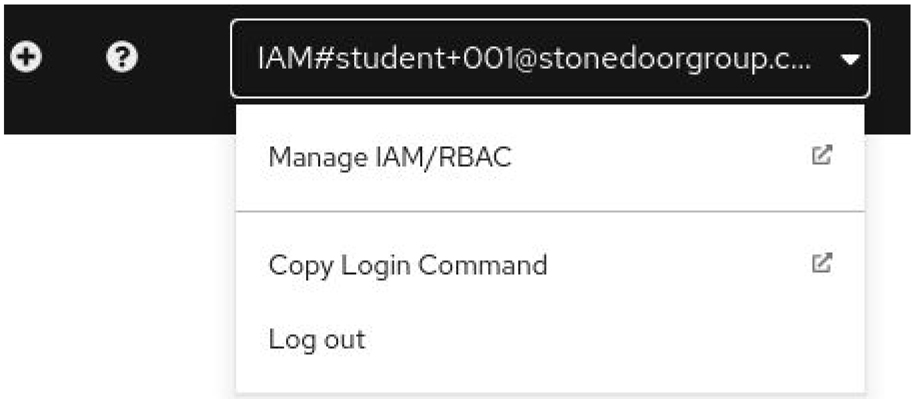


1.1 3Scale Installation Instructions

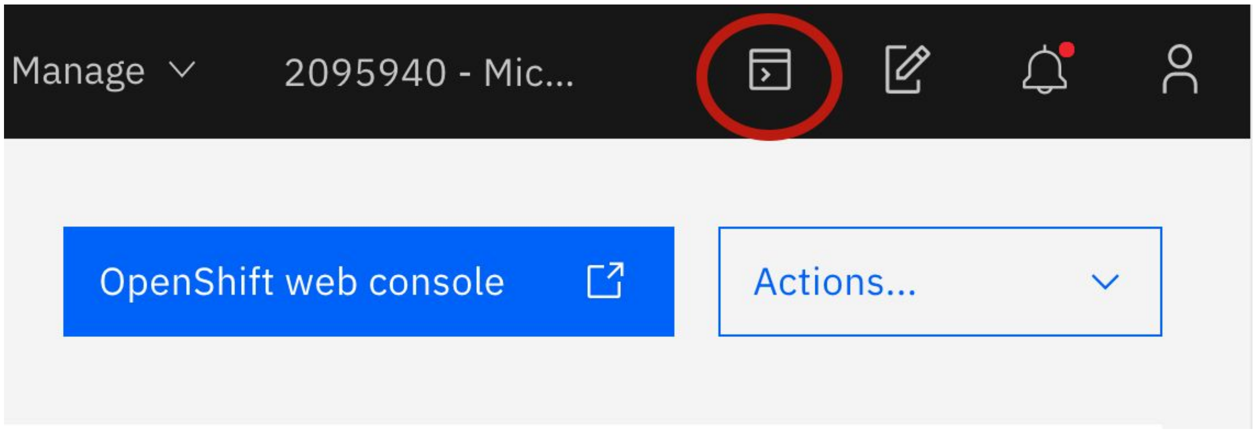
Follow these steps to install 3Scale API Management on OpenShift 4.x:

1. Using the account assigned to you, **login** to your OpenShift management console (GUI).

2. Obtain the command and token needed to login to the IBM Cloud Shell CLI from the 'Copy Login Command' option present in the user menu at the top right corner of the GUI):



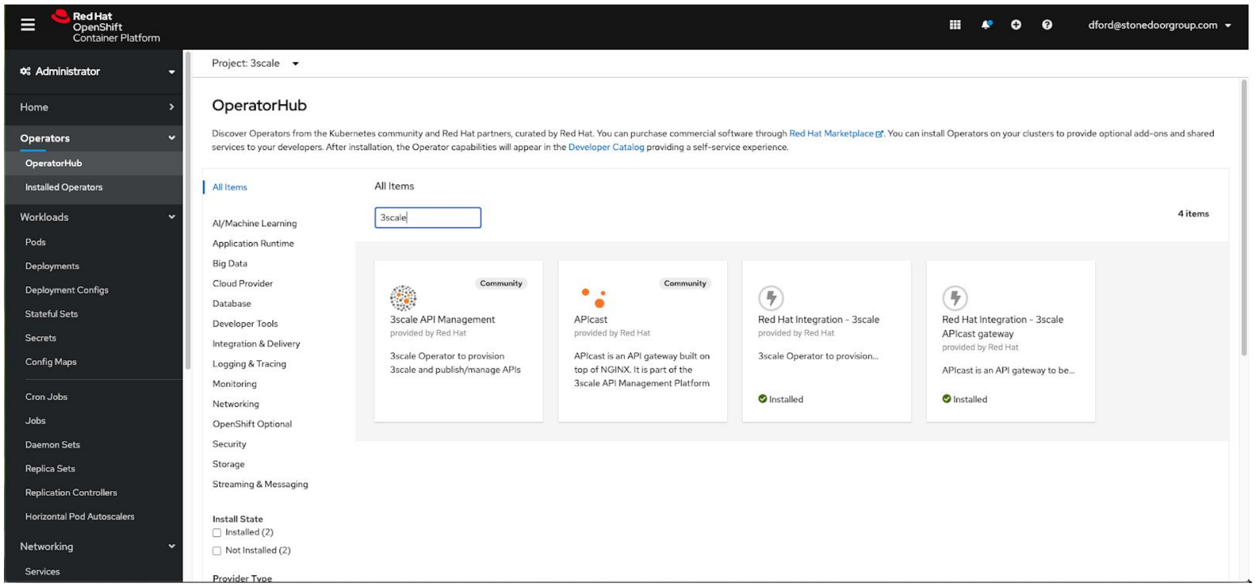
3. **Login** to the CLI using the IBM Cloud Shell terminal (which is started by clicking the terminal icon in the top bar, fourth from the far right):



Verify that the CLI login in the IBM Cloud Shell terminal worked by running the following commands in the same terminal:

```
$ oc whoami
IAM#<your_email>
$ oc project <firstname_lastname>-3scale
Now using project "<firstname-lastname>-3scale" on server
"https://c114-e.us-south.containers.cloud.ibm.com:32761".
```

4. To the left menu, select Operators -> Operator Hub, **select** the “Provider Type” of Red hat, then **search** for “3scale”. **Select** Red Hat Integration - 3scale:



OperatorHub

Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. You can purchase commercial software through Red Hat Marketplace. You can install Operators on your clusters to provide optional add-ons and shared services to your developers. After installation, the Operator capabilities will appear in the Developer Catalog providing a self-service experience.

All Items — 3scale — 4 items

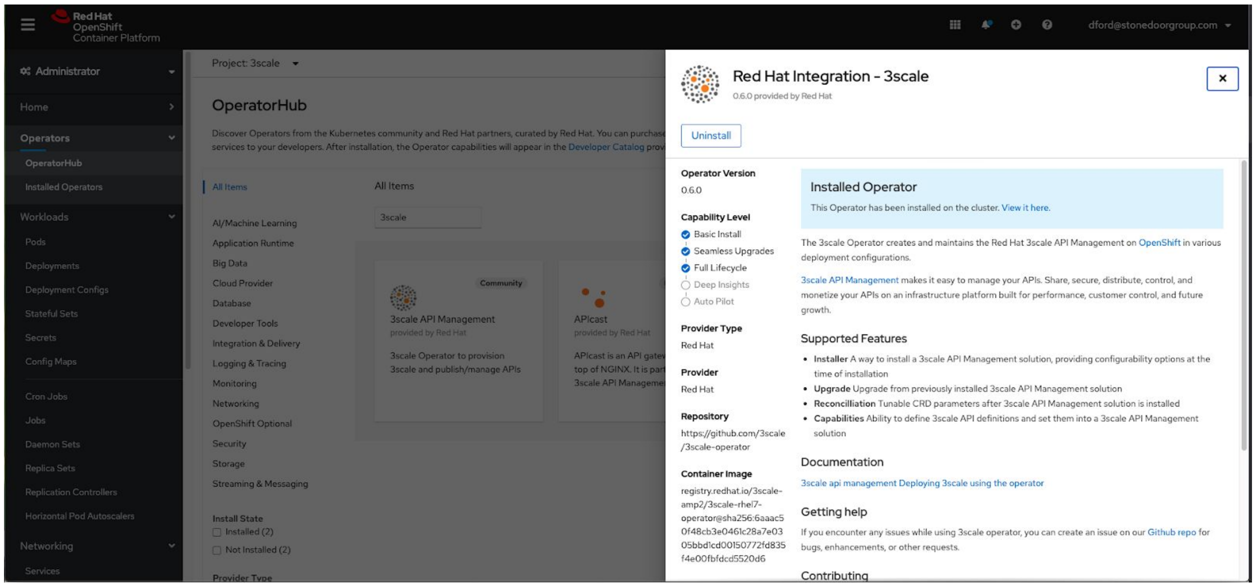
3scale API Management (Community) provided by Red Hat — 3scale Operator to provision 3scale and publish/manage APIs

APIcast (Community) provided by Red Hat — APIcast is an API gateway built on top of NGINX. It is part of the 3scale API Management Platform

Red Hat Integration - 3scale provided by Red Hat — 3scale Operator to provision... — Installed

Red Hat Integration - 3scale APIcast gateway provided by Red Hat — APIcast is an API gateway to be... — Installed

5. On the Operator Install screen, **click** “Install”; you will be taken to the Create Operator Subscription screen:



Red Hat Integration - 3scale
0.6.0 provided by Red Hat

Operator Version: 0.6.0

Capability Level: ✓ Basic Install, ✓ Seamless Upgrades, ✓ Full Lifecycle, Deep Insights, Auto Pilot

Provider Type: Red Hat

Provider: Red Hat

Repository: https://github.com/3scale/3scale-operator

Container Image: registry.redhat.io/3scale-amp2/3scale-rhel7-operator@sha256:6aaac50f48cb3e0461c28a7e0305bbd1cd00150772fd835f4e00fbfdcd5520d6

Installed Operator

This Operator has been installed on the cluster. View it here.

The 3scale Operator creates and maintains the Red Hat 3scale API Management on OpenShift in various deployment configurations.

3scale API Management makes it easy to manage your APIs. Share, secure, distribute, control, and monetize your APIs on an infrastructure platform built for performance, customer control, and future growth.

Supported Features

- **Installer** A way to install a 3scale API Management solution, providing configurability options at the time of installation
- **Upgrade** Upgrade from previously installed 3scale API Management solution
- **Reconcilliation** Tunable CRD parameters after 3scale API Management solution is installed
- **Capabilities** Ability to define 3scale API definitions and set them into a 3scale API Management solution

Documentation

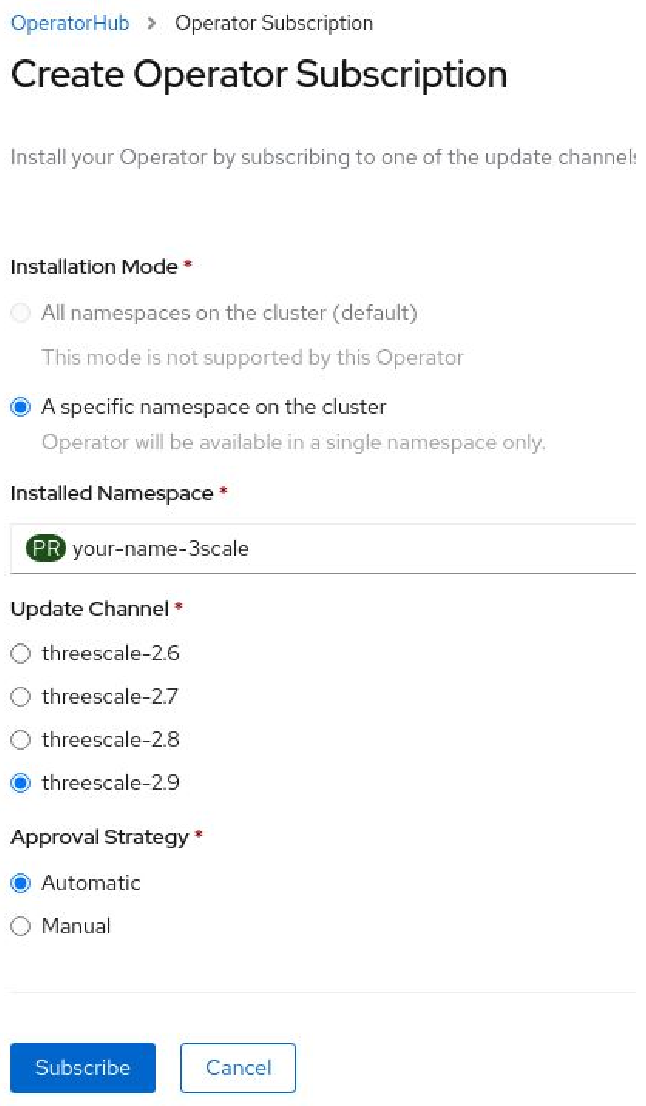
3scale api management Deploying 3scale using the operator

Getting help

If you encounter any issues while using 3scale operator, you can create an issue on our Github repo for bugs, enhancements, or other requests.

Contributing

6. On the Create Operator Subscription screen, **select** the project that was previously created for your account as the specific namespace to install into (<*firstname_lastname*>-3scale), then **click** Subscribe:



OperatorHub > Operator Subscription

Create Operator Subscription

Install your Operator by subscribing to one of the update channel

Installation Mode *

- All namespaces on the cluster (default)
  This mode is not supported by this Operator
- [✓ A specific namespace on the cluster]
  Operator will be available in a single namespace only.

Installed Namespace *

PR your-name-3scale

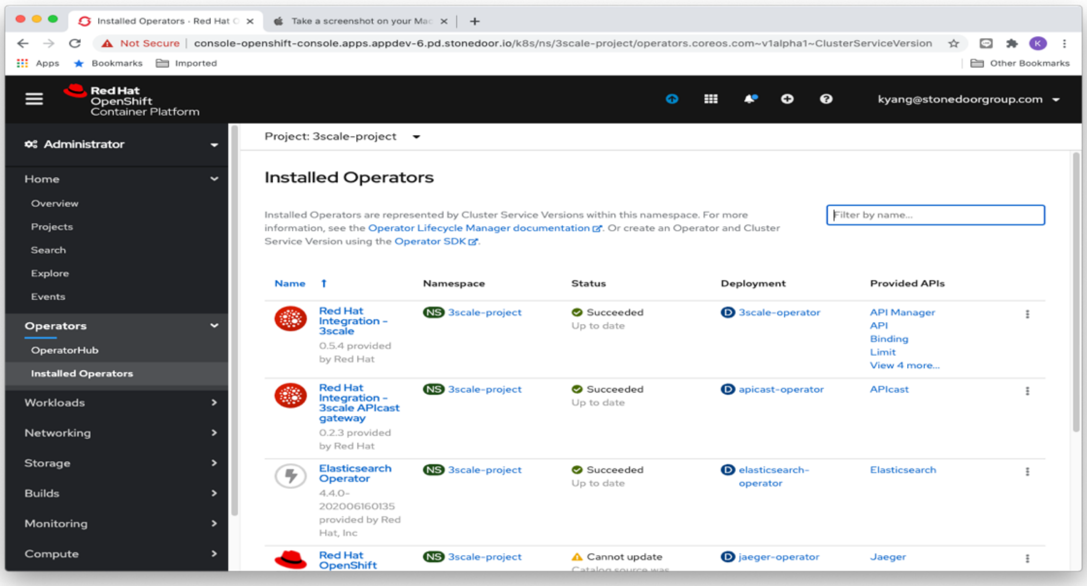
Update Channel *

- threescale-2.6
- threescale-2.7
- threescale-2.8
- [✓ threescale-2.9]

Approval Strategy *

- [✓ Automatic]
- Manual

7. From the “Installed Operators” page, **monitor** the Status of the 3scale operator until it indicates it has Succeeded and is “up to date”:



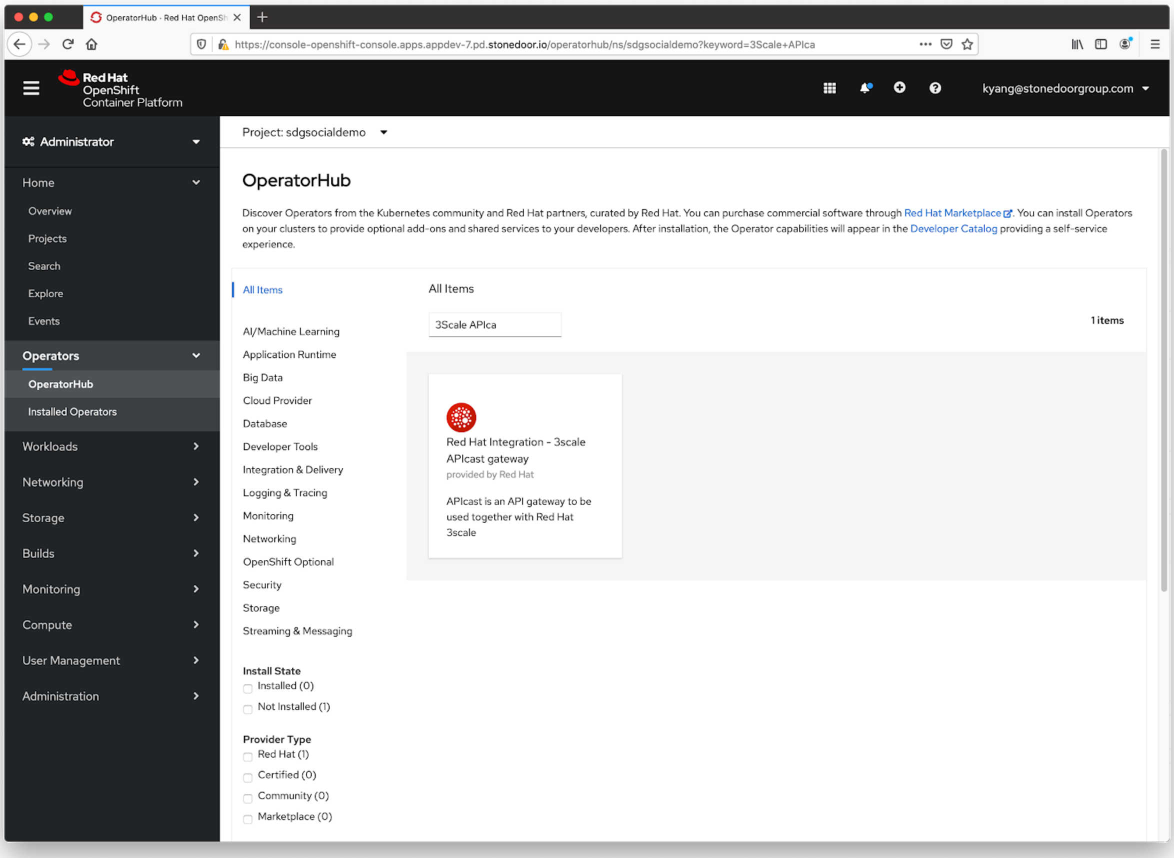
Project: 3scale-project

Installed Operators

Installed Operators are represented by Cluster Service Versions within this namespace. For more information, see the Operator Lifecycle Manager documentation. Or create an Operator and Cluster Service Version using the Operator SDK.

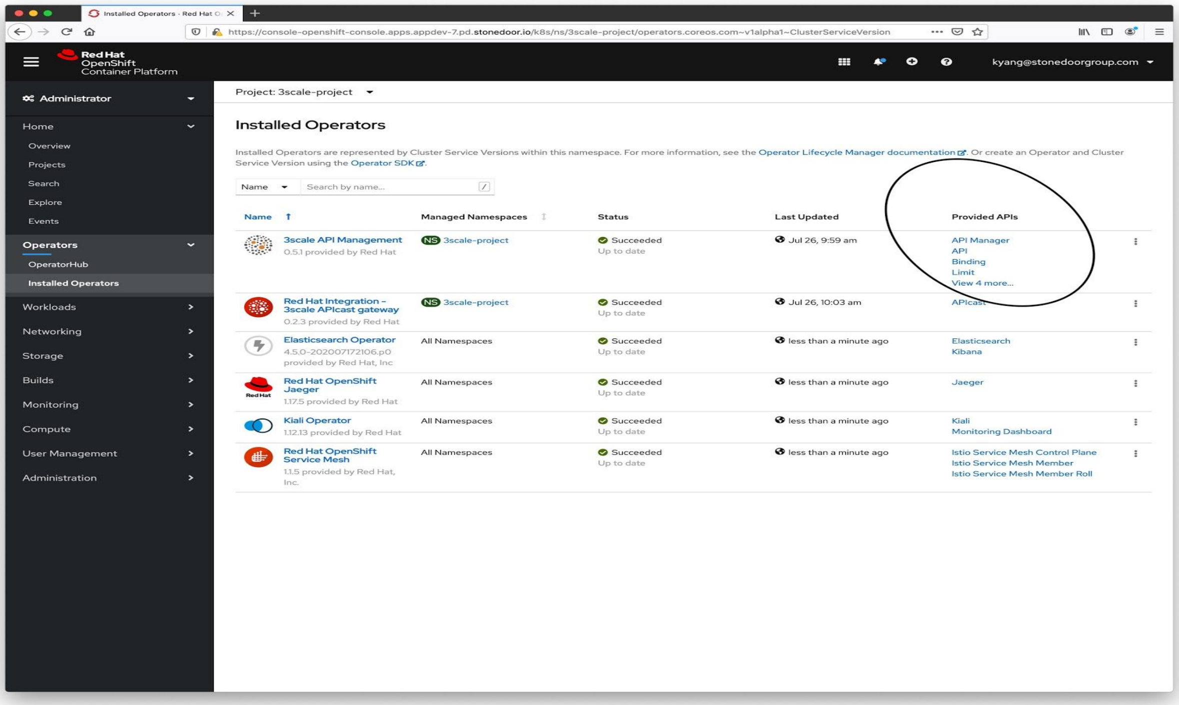
Name	Namespace	Status	Deployment	Provided APIs
Red Hat Integration - 3scale 0.5.4 provided by Red Hat	NS 3scale-project	Succeeded Up to date	D 3scale-operator	API Manager API Binding Limit View 4 more...
Red Hat Integration - 3scale APIcast gateway 0.2.3 provided by Red Hat	NS 3scale-project	Succeeded Up to date	D apicast-operator	APIcast
Elasticsearch Operator 4.4.0-202006160135 provided by Red Hat, Inc	NS 3scale-project	Succeeded Up to date	D elasticsearch-operator	Elasticsearch
Red Hat OpenShift	NS 3scale-project	Cannot update	D jaeger-operator	Jaeger

8. **Return** to the Operators -> OperatorHub screen, and **select** the “Red Hat Integration 3scale APIcast gateway”. Install into the same personal namespace, again allowing the default automatic approval strategy:


OperatorHub · Red Hat OpenSh
https://console-openshift-console.apps.appdev-7.pd.stonedoor.io/operatorhub/ns/sdgsocialdemo?keyword=3Scale+APIca
Red Hat OpenShift Container Platform
kyang@stonedoorgroup.com
Administrator
Home
Overview
Projects
Search
Explore
Events
Operators
OperatorHub
Installed Operators
Workloads
Networking
Storage
Builds
Monitoring
Compute
User Management
Administration
Project: sdgsocialdemo
OperatorHub
Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. You can purchase commercial software through Red Hat Marketplace. You can install Operators on your clusters to provide optional add-ons and shared services to your developers. After installation, the Operator capabilities will appear in the Developer Catalog providing a self-service experience.
All Items
AI/Machine Learning
Application Runtime
Big Data
Cloud Provider
Database
Developer Tools
Integration & Delivery
Logging & Tracing
Monitoring
Networking
OpenShift Optional
Security
Storage
Streaming & Messaging
Install State
Installed (0)
Not Installed (1)
Provider Type
Red Hat (1)
Certified (0)
Community (0)
Marketplace (0)
All Items
3Scale APIca
1 items
Red Hat Integration - 3scale APIcast gateway
provided by Red Hat
APIcast is an API gateway to be used together with Red Hat 3scale

When the installation completes, you are taken back to the Installed Operators page. **Monitor** the progress, waiting until it has succeeded and is up to date.

9. The installed 3scale operators allow for the creation of 3scale component instances via CRDs. Create an APIManager instance by **clicking** on the "API Manager" link in the far right column labeled "Provided APIs". **Click** on the "Create APIManager" button, this will take you to a YAML editing window to customize your installation:



Installed Operators

Installed Operators are represented by Cluster Service Versions within this namespace. For more information, see the Operator Lifecycle Manager documentation. Or create an Operator and Cluster Service Version using the Operator SDK.

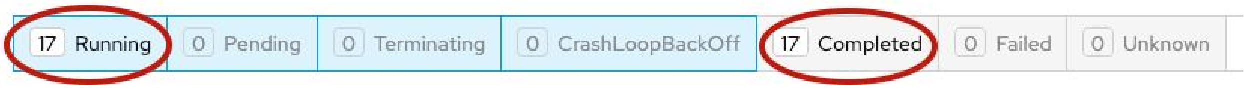
Name	Managed Namespaces	Status	Last Updated	Provided APIs
3scale API Management 0.5.1 provided by Red Hat	NS 3scale-project	Succeeded Up to date	Jul 26, 9:59 am	API Manager API Binding Limit View 4 more...
Red Hat Integration - 3scale APIcast gateway 0.2.3 provided by Red Hat	NS 3scale-project	Succeeded Up to date	Jul 26, 10:03 am	APIcast
Elasticsearch Operator 4.5.0-202007172106.p0 provided by Red Hat, Inc	All Namespaces	Succeeded Up to date	less than a minute ago	Elasticsearch Kibana
Red Hat OpenShift Jaeger 1.17.5 provided by Red Hat	All Namespaces	Succeeded Up to date	less than a minute ago	Jaeger
Kiali Operator 1.12.13 provided by Red Hat	All Namespaces	Succeeded Up to date	less than a minute ago	Kiali Monitoring Dashboard
Red Hat OpenShift Service Mesh 1.1.5 provided by Red Hat, Inc.	All Namespaces	Succeeded Up to date	less than a minute ago	Istio Service Mesh Control Plane Istio Service Mesh Member Istio Service Mesh Member Roll

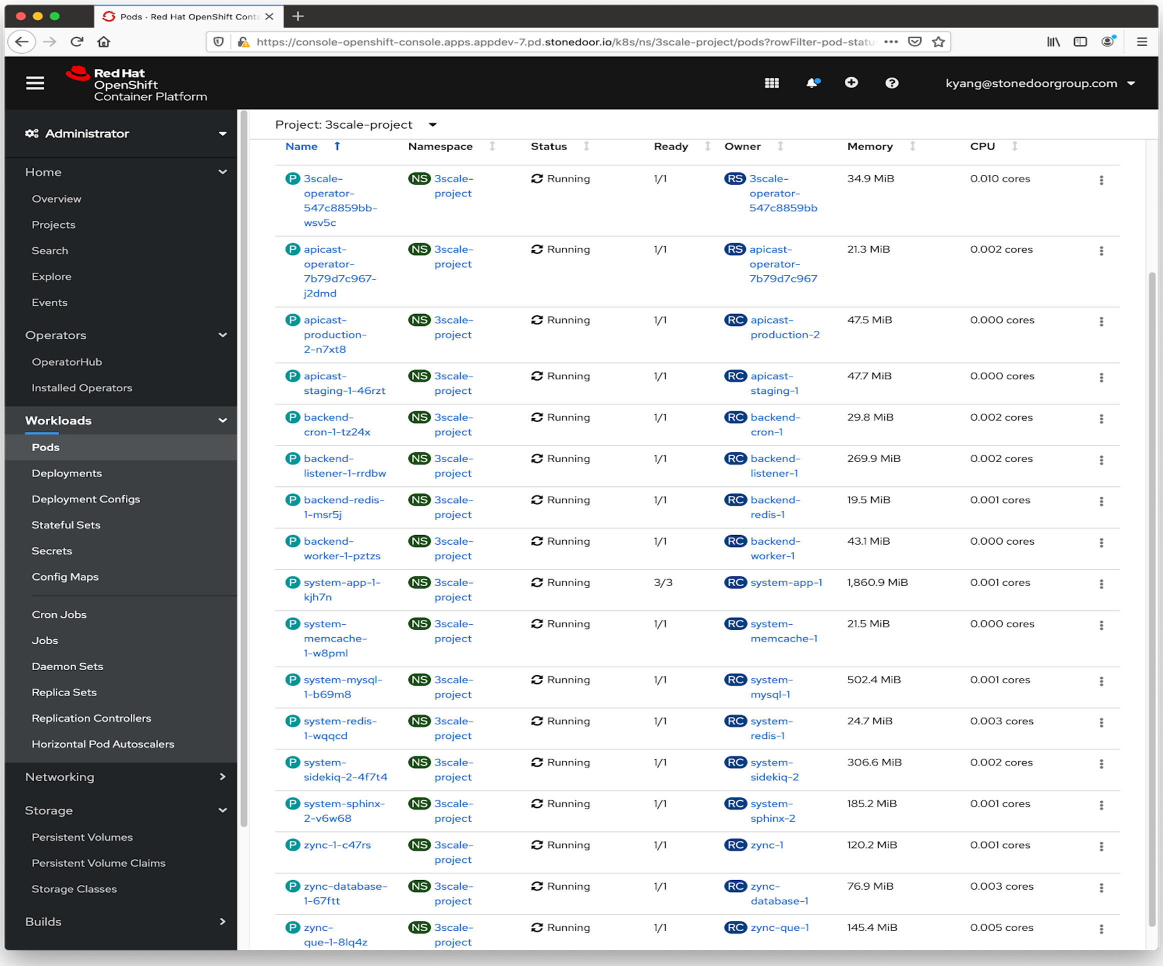
10. In the "Create APIManager" window, you are presented with a YAML template for this CRD. **Edit** to match the following, replacing the wildcardDomain value represented by the <*firstname-lastname*> placeholder with the value matching your assigned project:

```yaml
apiVersion: apps.3scale.net/v1alpha1
kind: APIManager
metadata:
  name: example-apimanager
  namespace: <firstname-lastname>-3scale
spec:
  resourceRequirementsEnabled: false
  system:
    fileStorage:
      persistentVolumeClaim:
        storageClassName: managed-nfs-storage
  wildcardDomain: <firstname-lastname>.3scale-lab.pd.stonedoor.io
```

Once you have completed your edits, and double checked your work, **click** Create, and the operator will create all the associated resources (42 objects in total).

11. **Navigate** to the Workload-->Pods menu from the OpenShift GUI and check all Pods are up and running correctly (this might take up to 15 minutes depending on cluster load). You will see deployer Pods come up first, then exit when their corresponding workload Pods are started. In total the number should stabilize at 17 pods in running state, and 17 in completed:

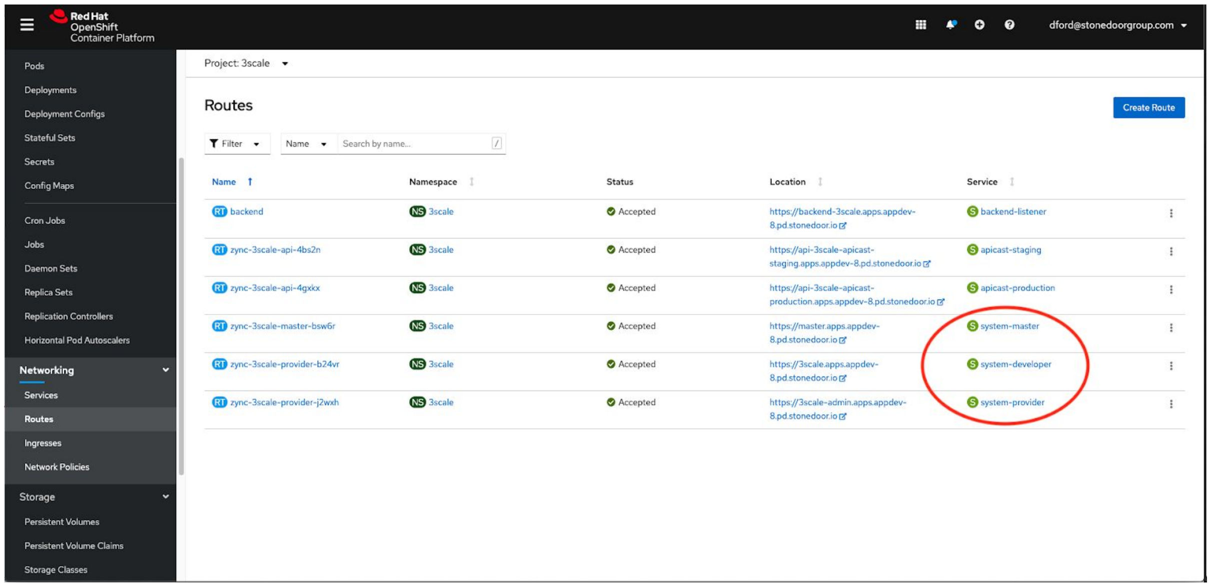




Project: 3scale-project

Name	Namespace	Status	Ready	Owner	Memory	CPU
P 3scale-operator-547c8859bb-wsv5c	NS 3scale-project	Running	1/1	RS 3scale-operator-547c8859bb	34.9 MiB	0.010 cores
P apicast-operator-7b79d7c967-j2dmd	NS 3scale-project	Running	1/1	RS apicast-operator-7b79d7c967	21.3 MiB	0.002 cores
P apicast-production-2-n7xt8	NS 3scale-project	Running	1/1	RC apicast-production-2	47.5 MiB	0.000 cores
P apicast-staging-1-46rzt	NS 3scale-project	Running	1/1	RC apicast-staging-1	47.7 MiB	0.000 cores
P backend-cron-1-tz24x	NS 3scale-project	Running	1/1	RC backend-cron-1	29.8 MiB	0.002 cores
P backend-listener-1-rrdbw	NS 3scale-project	Running	1/1	RC backend-listener-1	269.9 MiB	0.002 cores
P backend-redis-1-msr5j	NS 3scale-project	Running	1/1	RC backend-redis-1	19.5 MiB	0.001 cores
P backend-worker-1-pztzs	NS 3scale-project	Running	1/1	RC backend-worker-1	43.1 MiB	0.000 cores
P system-app-1-kjh7n	NS 3scale-project	Running	3/3	RC system-app-1	1,860.9 MiB	0.001 cores
P system-memcache-1-w8pml	NS 3scale-project	Running	1/1	RC system-memcache-1	21.5 MiB	0.000 cores
P system-mysql-1-b69m8	NS 3scale-project	Running	1/1	RC system-mysql-1	502.4 MiB	0.001 cores
P system-redis-1-wqqcd	NS 3scale-project	Running	1/1	RC system-redis-1	24.7 MiB	0.003 cores
P system-sidekiq-2-4f7t4	NS 3scale-project	Running	1/1	RC system-sidekiq-2	306.6 MiB	0.002 cores
P system-sphinx-2-v6w68	NS 3scale-project	Running	1/1	RC system-sphinx-2	185.2 MiB	0.001 cores
P zync-1-c47rs	NS 3scale-project	Running	1/1	RC zync-1	120.2 MiB	0.001 cores
P zync-database-1-67ftt	NS 3scale-project	Running	1/1	RC zync-database-1	76.9 MiB	0.003 cores
P zync-que-1-8lq4z	NS 3scale-project	Running	1/1	RC zync-que-1	145.4 MiB	0.005 cores

12. Once all expected Pods are running, examine the network routes which were created to expose access to the 3scale resources. From the OpenShift administration console menu on the left, **navigate** to “Networking -> Routes” and **confirm** that the following routes have been created:



Routes

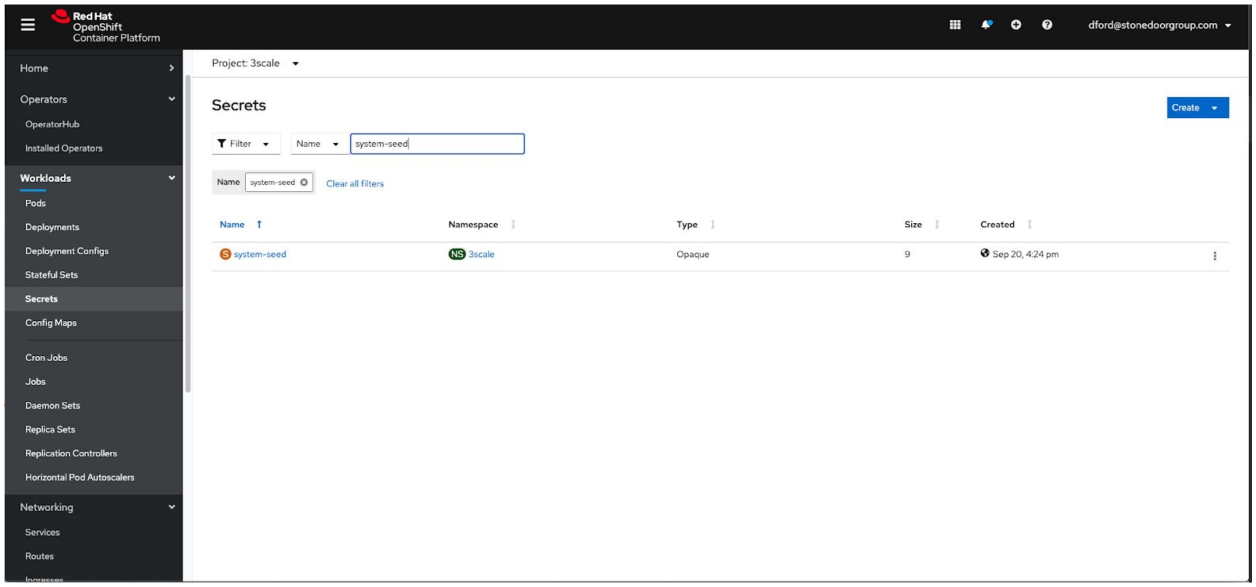
Name	Namespace	Status	Location	Service
RT backend	NS 3scale	Accepted	https://backend-3scale.apps.appdev-8.pd.stonedoor.io	S backend-listener
RT zync-3scale-api-4bs2n	NS 3scale	Accepted	https://api-3scale-apicast-staging.apps.appdev-8.pd.stonedoor.io	S apicast-staging
RT zync-3scale-api-4gxkx	NS 3scale	Accepted	https://api-3scale-apicast-production.apps.appdev-8.pd.stonedoor.io	S apicast-production
RT zync-3scale-master-bsw6r	NS 3scale	Accepted	https://master.apps.appdev-8.pd.stonedoor.io	S system-master
RT zync-3scale-provider-b24vr	NS 3scale	Accepted	https://3scale.apps.appdev-8.pd.stonedoor.io	S system-developer
RT zync-3scale-provider-j2wxh	NS 3scale	Accepted	https://3scale-admin.apps.appdev-8.pd.stonedoor.io	S system-provider

13. Now that all 3Scale software, storage, and network components have been deployed successfully in OpenShift, you may access the 3Scale interfaces. All routes will be under a `<firstname-lastname>.3scale-lab.pd.stonedoor.io` domain. There are a total of 3 interfaces:

    - **Master** - used for organization to manage all organization's API management - https://master.<*name*>.3scale-lab.pd.stonedoor.io (forwarding to the system-master service)
    - **Admin** - used for https://3scale-admin.<*name*>.3scale-lab.pd.stonedoor.io (forwarding to the system-provider service)
    - **Developer Portal** - this is the default external customer facing portal that your developers will use to get self-service access to keys and also access your documentation - https://3scale.<*name*>.3scale-lab.pd.stonedoor.io (forwarding to the system-developer service)

14. Upon installation, 3Scale created master usernames with random generated passwords. The credentials will be needed in a later step, and can be viewed either via the GUI or CLI.

    From the GUI, **navigate** to Workloads->Secrets and **click** on system-seed, this setting contains the encoded username/password for accessing master, admin portal.



Name	Namespace	Type	Size	Created
system-seed	3scale	Opaque	9	Sep 20, 4:24 pm

From within the system-seed screen, **click** the “reveal values” and then either record the values, or leave this browser tab open.

Alternatively, from the IBM Cloud terminal CLI, get the decoded values by running the following command, and then record them for future reference:

```
$ oc get secret/system-seed -n <firstname-lastname>-3scale -o json | jq '.data
| map_values(@base64d)'
{
  "ADMIN_ACCESS_TOKEN": "KA3Ug1Lb6NvT7Qhz",
  "ADMIN_EMAIL": "",
  "ADMIN_PASSWORD": "thdaNQzr",
  "ADMIN_USER": "admin",
  "MASTER_ACCESS_TOKEN": "Vfsr5heg",
  "MASTER_DOMAIN": "master",
  "MASTER_PASSWORD": "yJ59SePl",
  "MASTER_USER": "master",
  "TENANT_NAME": "3scale"
}
```

1.2 Conclusion

You now have a running 3Scale API environment within OpenShift. In the following sections, we will describe how to setup 3rd party applications to consume WeNote APIs through the 3scale gateway.

1.3 Interfacing 3Scale with Service Mesh on OpenShift

One of the benefits of installing 3Scale on OpenShift is that it enables 3Scale to leverage Service Mesh, a full application observability platform. Red Hat's Service Mesh Operator is based on Istio, along with required components for observability, including Prometheus, Grafana, Kiali, Jaeger.

By interfacing with Service Mesh, the 3Scale administrator will be able to observe both the internal API access of the application and the external developers accessing the application via 3Scale APICast.

1. The Service Mesh control plane has already been installed into the istio-system project, and will be shared by all participants. Do NOT make ANY changes to the control plane other than to enroll your namespace following the directions below.

2. The cluster is configured so that projects must opt-in to use the already deployed Service Mesh. In the IBM Cloud Shell terminal make your application project active, and enroll it for Service Mesh use:

```
$ oc project <firstname-lastname>-wenote
Now using project "<firstname_lastname>-wenote" on server
"https://c114-e.us-south.containers.cloud.ibm.com:32761".
$ oc patch ServiceMeshMemberRoll/default --type='json' -p '[{"op": "add",
"path": "/spec/members/-", "value": "<firstname-lastname>-wenote"}]' -n
istio-system
servicemeshmemberroll.maistra.io/default patched
```

3. Verify that you are listed in the configuredMembers status field:

```
$ oc get ServiceMeshMemberRoll/default -o
jsonpath='{.status.configuredMembers}' -n istio-system
[... <firstname_lastname>-wenote <other_name>-wenote ...]
```

2.0 Install Sample Application Application on OpenShift

Now that we have OpenShift, ServiceMesh and 3Scale running, the next steps are to import the WeNote application into OpenShift.

2.1 WeNote Installation on OpenShift

The sample WeNote application is available here:

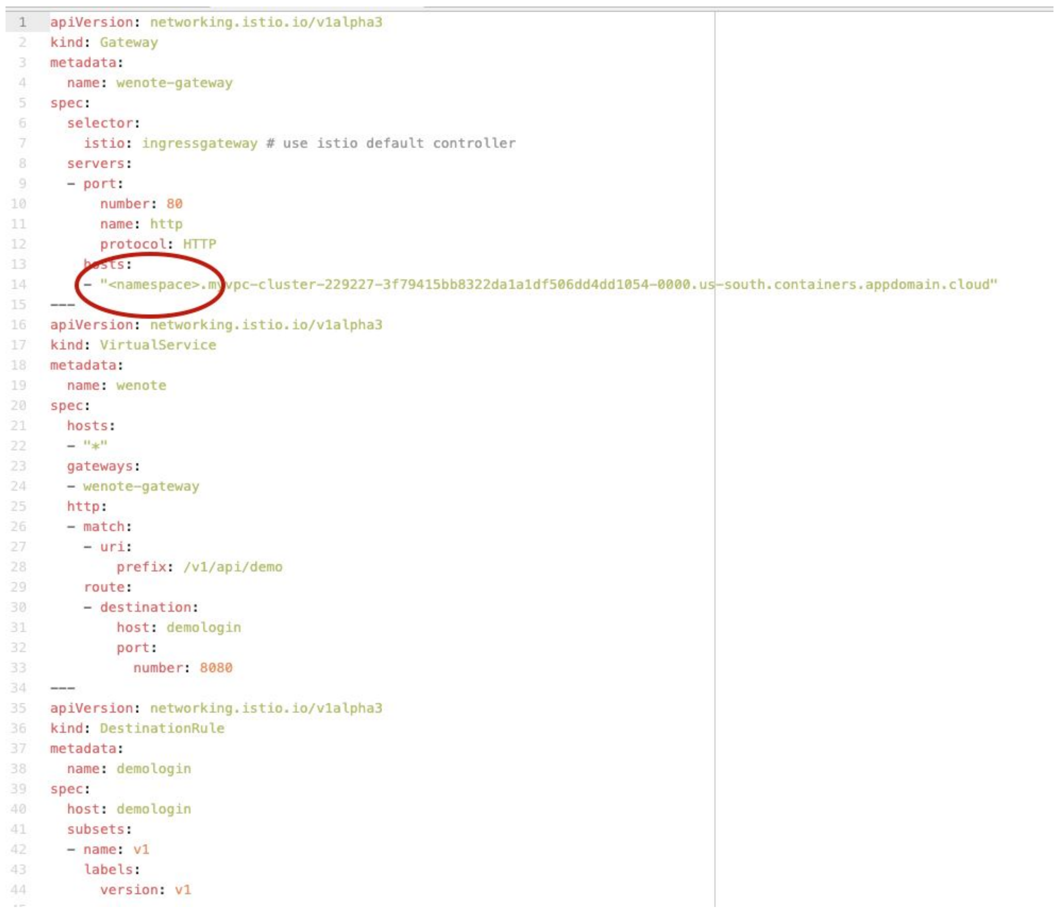
`https://github.com/zhejingl/demologin`

The following steps install WeNote on OpenShift and make WeNote available for observability using ServiceMesh.

1. Install the WeNote application into your project by running the following commands from the IBM Cloud Shell terminal window:

```
$ cd; git clone https://github.com/zhejingl/demologin.git
Cloning into 'demologin'...
remote: Enumerating objects: 62, done.
remote: Counting objects: 100% (62/62), done.
remote: Compressing objects: 100% (40/40), done.
remote: Total 62 (delta 14), reused 56 (delta 8), pack-reused 0
Unpacking objects: 100% (62/62), done
$ cd demologin/
$ oc apply -f sdgdemo.yaml
service/demologin created
serviceaccount/sdgsocialdemo-demologin created
deployment.apps/demologin-v1 created
$ oc get pods
NAME                            READY   STATUS    RESTARTS   AGE
demologin-v1-6bc9bd597b-sqm6f   2/2     Running   0          40s
```

2. Create the Service Mesh specific resources for the application. **Edit** the `demogateway.yaml` (the Vi and Nano editors are both available in the Cloud Shell). **Change** the placeholder value <*namespace*> to your assigned, personal namespace <*firstname-lastname*-wenote>:



```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: wenote-gateway
spec:
  selector:
    istio: ingressgateway # use istio default controller
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - "<namespace>.myvpc-cluster-229227-3f79415bb8322da1a1df506dd4dd1054-0000.us-south.containers.appdomain.cloud"
---
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: wenote
spec:
  hosts:
  - "*"
  gateways:
  - wenote-gateway
  http:
  - match:
    - uri:
        prefix: /v1/api/demo
    route:
    - destination:
        host: demologin
        port:
          number: 8080
---
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: demologin
spec:
  host: demologin
  subsets:
  - name: v1
    labels:
      version: v1
```

Optionally, you can perform and verify the edit as follows:

```
$ sed -i 's/<namespace>/<firstname-lastname>-wenote/' demogateway.yaml
$ cat demogateway.yaml
...output omitted...
```

3. Using the edited and verified file, create the objects in your assigned namespace:

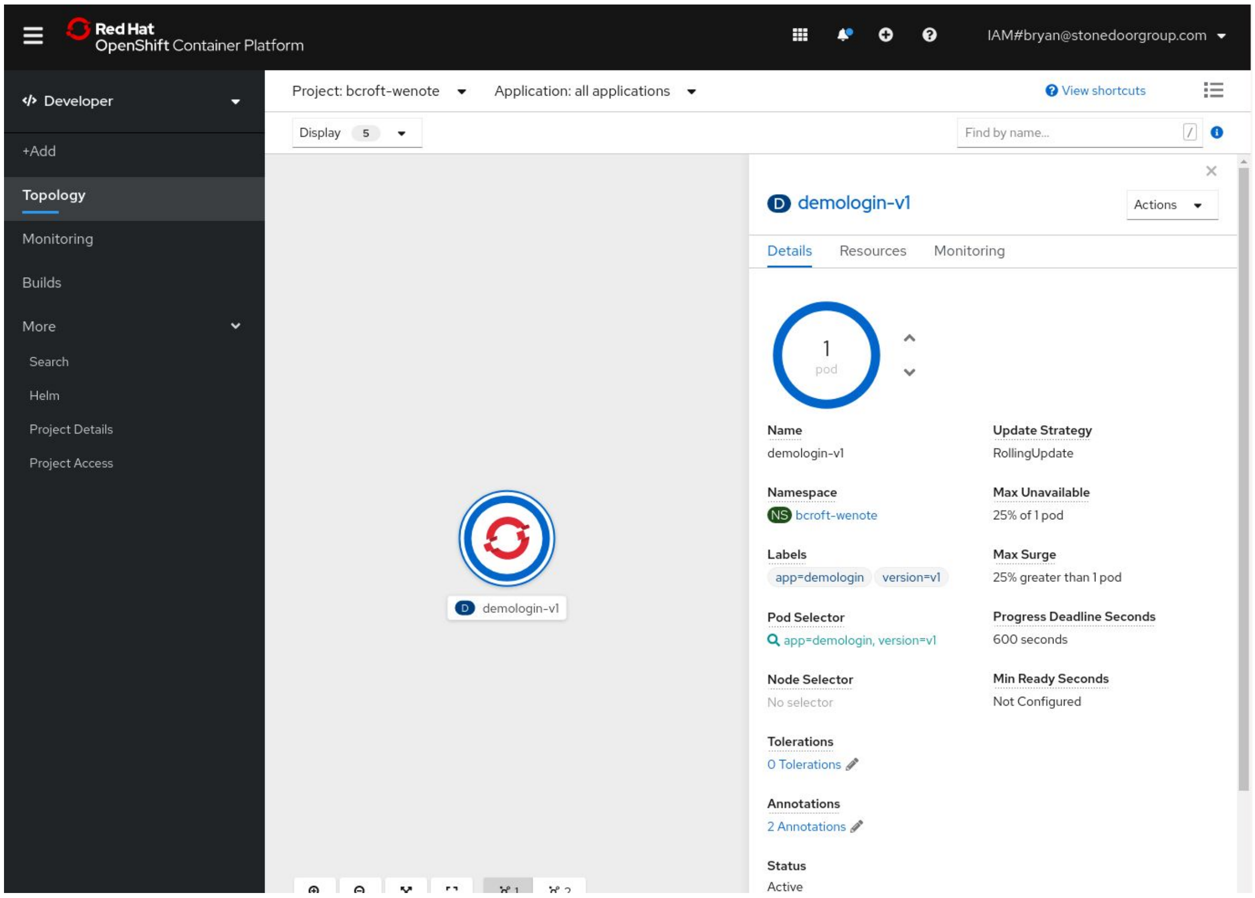
```
$ oc apply -f demogateway.yaml
destinationrule.networking.istio.io/demologin created
gateway.networking.istio.io/sdgsocialdemo-gateway created
virtualservice.networking.istio.io/bookinfo created
```

4. Now that the route and gateway to Service Mesh is established, test your connection by using Curl from the command line; you should receive a JSON formatted token in return:

```
$ BASE=$(oc get routes -n openshift-ingress -o jsonpath='{..routerCanonicalHostname}')
$ echo $BASE
myvpc-cluster-229227-3f79415bb8322da1a1df506dd4dd1054-0000.us-south.containers.
appdomain.cloud
```

```
$ curl -k -X POST
http://<firstname-lastname>-wenote.$BASE/v1/api/demo/social/login -d'{}' -H
'Content-Type: application/json'
{"token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJleHAiOjE2MDM1NDkyMDB9.ycdn3jA
JWW2v1HmO-3fBwwf00lxetLrCLLqYju66BnVtNwevAEpzEK63G8r-VacaGLUq9XGdEPAh5a6HgQ2yPA
","enabled":[]}
```

5. From the GUI console, on the top left, **select** the “Developer” view. From the sidebar, **select** Topology, and for Project, select *<firstname_lastname>*-wenote. A single deployment named `demologin-v1` should be displayed. **Click** it to view the detailed information about the deployment and to verify that it is running:



Name
demologin-v1

Update Strategy
RollingUpdate

Namespace
NS bcroft-wenote

Max Unavailable
25% of 1 pod

Labels
app=demologin version=v1

Max Surge
25% greater than 1 pod

Pod Selector
app=demologin, version=v1

Progress Deadline Seconds
600 seconds

Node Selector
No selector

Min Ready Seconds
Not Configured

Tolerations
0 Tolerations

Annotations
2 Annotations

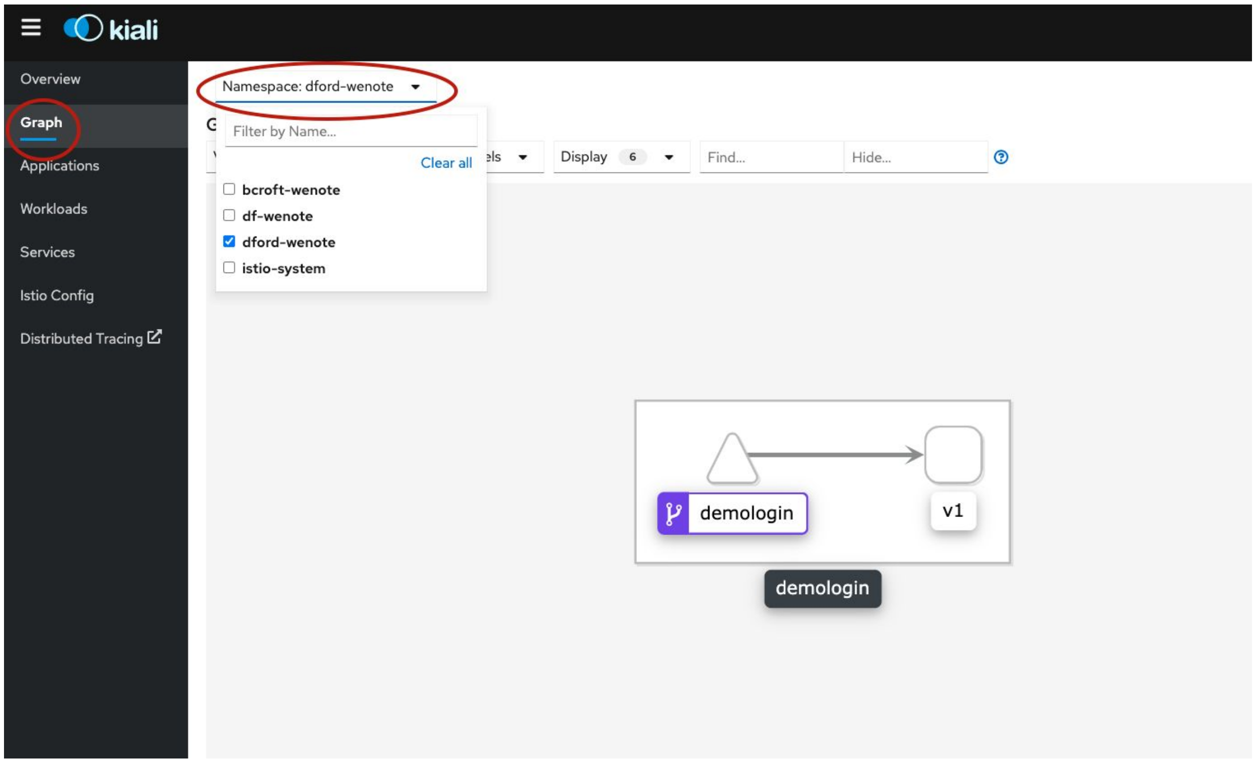
Status
Active

6. Determine the URL for the Kiali Dashboard by running the following query:

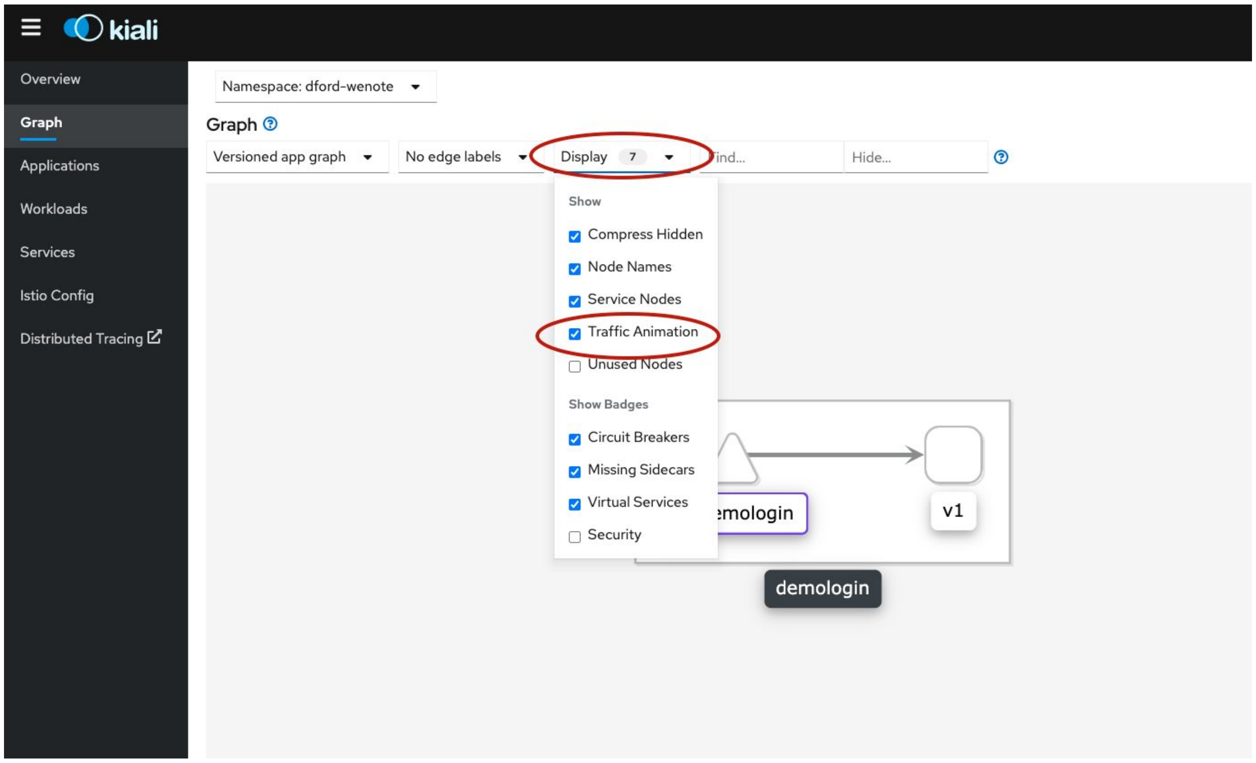
```
$ oc get route/kiali -n istio-system -o jsonpath='https://{.spec.host}{"\n"}'
https://kiali-istio-system.myvpc-cluster-...snip...us-south.containers.appdomai
n.cloud
```

7. **Open** the URL obtained in the previous step in your browser, and **click** “Login with your Openshift login”. You should see your application listed on the overview page (other participants applications will also be shown).

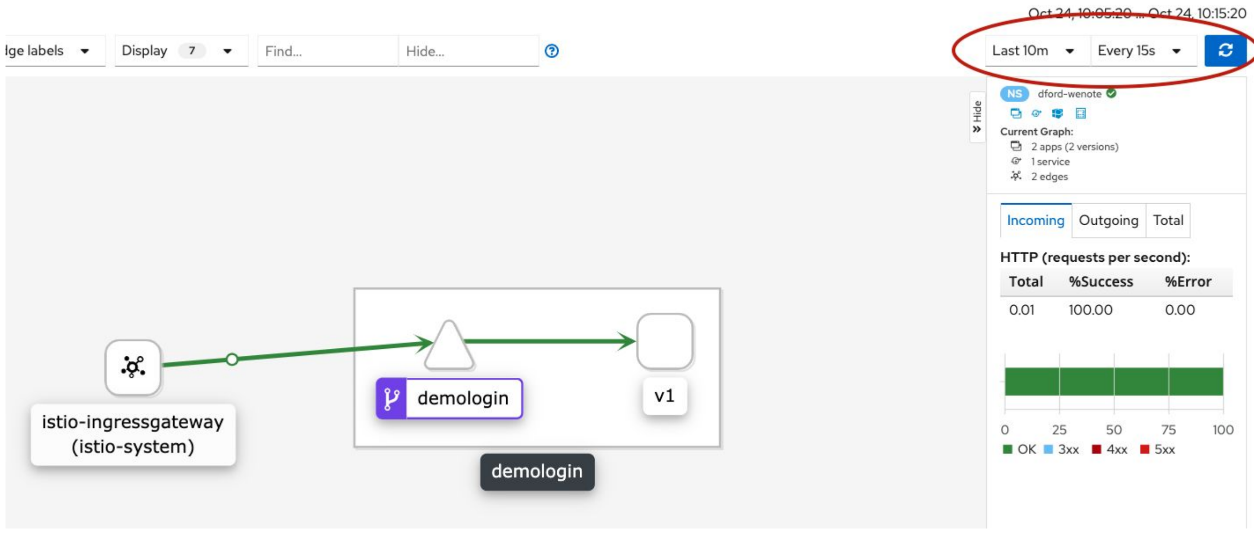
8. **Click** on “Graph” to the left of the Kiali dashboard, and **select** your namespace:



9. Now **select** the display dropdown and **select** Traffic Animation



10. You can adjust how long of a history you wish to show and a refresh frequency in the upper right hand corner - an animation should show the traffic through the Istio-IngressGateway to your service. (You can repeat the Curl command a few times to see more traffic):



Total	%Success	%Error
0.01	100.00	0.00

2.2 Conclusion

With the WeNote application installed, and linked to Service Mesh, it can now be added as an API in the 3Scale API Gateway.

3.0 Create 3Scale Corporate Site and Setup API Gateway

Now that we have WeNote running on OpenShift, we will configure 3Scale to serve a developer portal for WeNote. We will configure the developer front end portal, key self service, and demonstrate how a 3rd party developer or internal developer group can make calls to the 3Scale API that will broker the call to the core WeNote API.

3.1 Create A Development Group on 3Scale

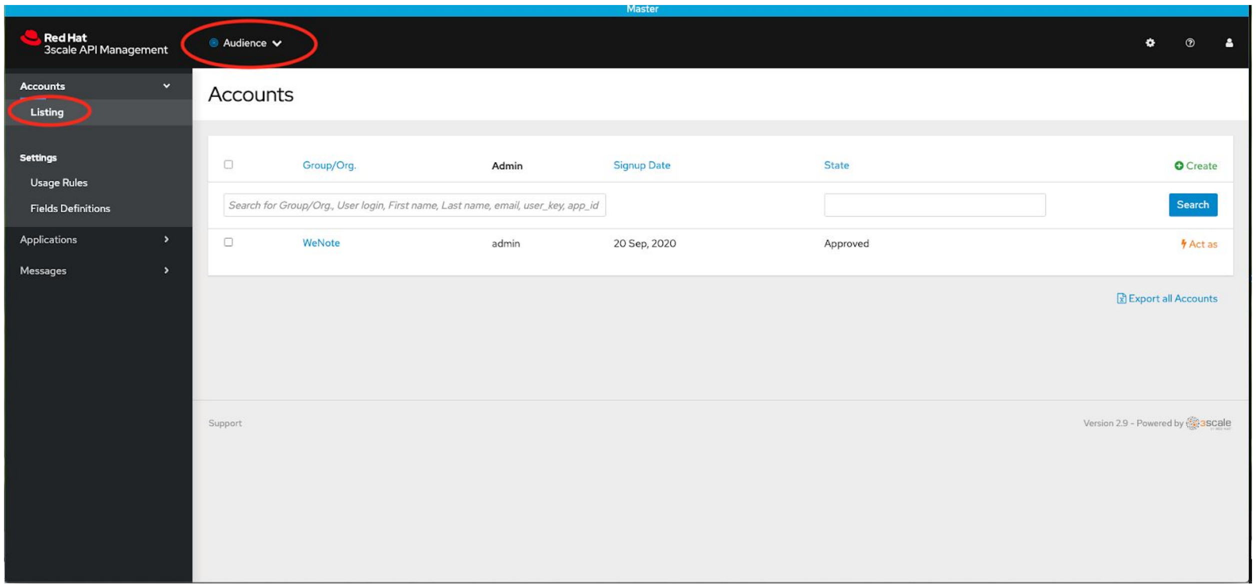
The first step in setting up 3Scale is to create a Development Group. Development Groups are the administrators for one or more applications in 3Scale. In order to successfully admin the WeNote application, we must first create a Development Group to manage 3Scale.

1. **Obtain** the 3scale master URL by either finding the route in your 3scale project from the GUI or running the following from your Cloud Shell:

```
$ oc get route -l zync.3scale.net/route-to=system-master -o
jsonpath='https://{..spec.host}{"\n"}' -n <firstname-lastname>-3scale
https://master.<firstname_lastname>.3scale-lab.pd.stonedoor.io
```

2. **Login** on to the master by opening your browser to the URL, with the MASTER_USER and MASTER_PASSWORD credentials obtained and recorded in the previous step (from the system-seed secret).

3. From top drop down, **select** Audience, then on the left sidebar, **select** “Accounts->Listing.”

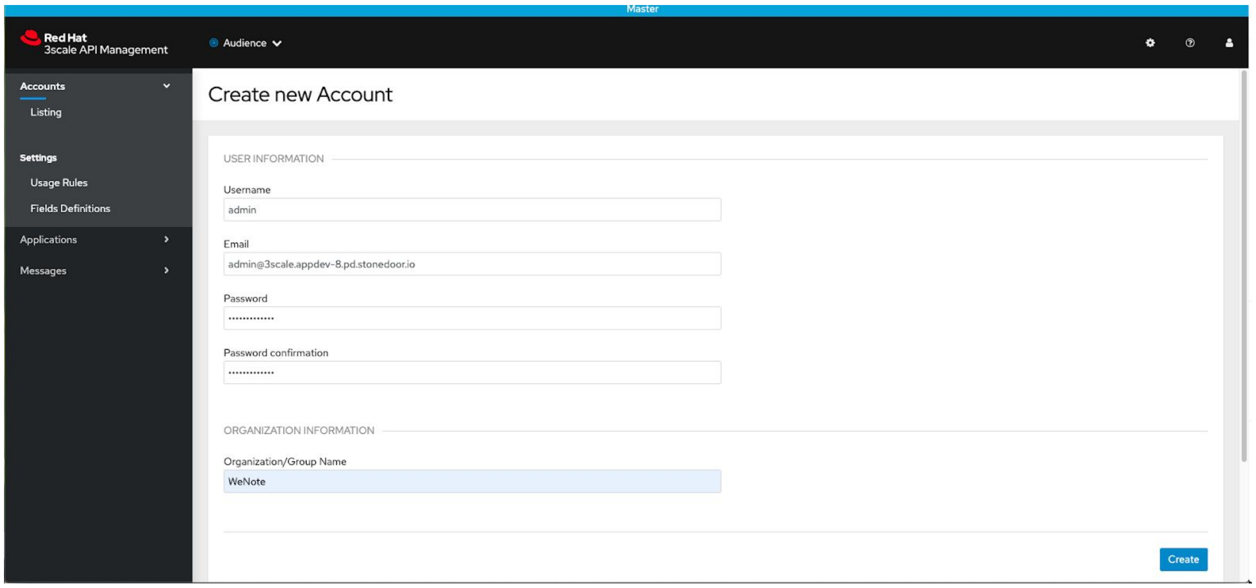


Accounts

	Group/Org.	Admin	Signup Date	State	Create
	Search for Group/Org., User login, First name, Last name, email, user_key, app_id				Search
	WeNote	admin	20 Sep, 2020	Approved	Act as

Export all Accounts

4. Create an administrator for the WeNote Development group by: **clicking** on "Create" and completing the form:
   - Username: **`admin`**
   - Email: **`admin@example.com`**
   - Password: **`adminpass`**
   - Organization/Group name: **`WeNote`**



Create new Account

USER INFORMATION

Username: admin

Email: admin@3scale.appdev-8.pd.stonedoor.io

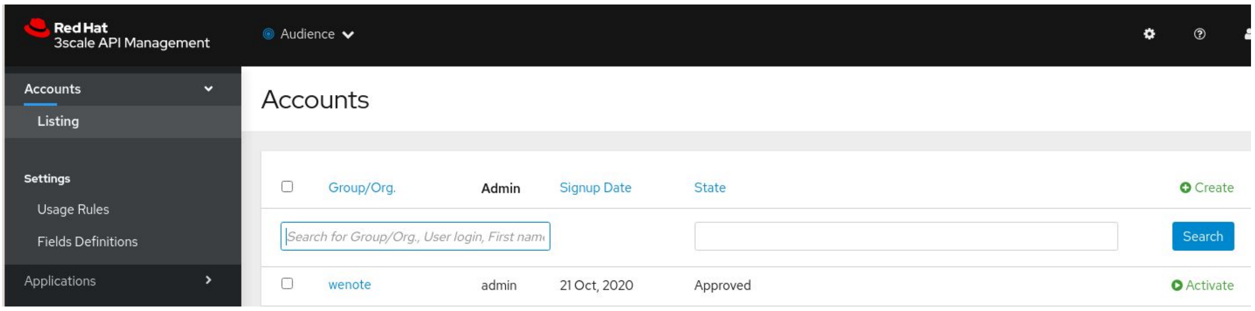
Password: ••••••••••••

Password confirmation: ••••••••••••

ORGANIZATION INFORMATION

Organization/Group Name: WeNote

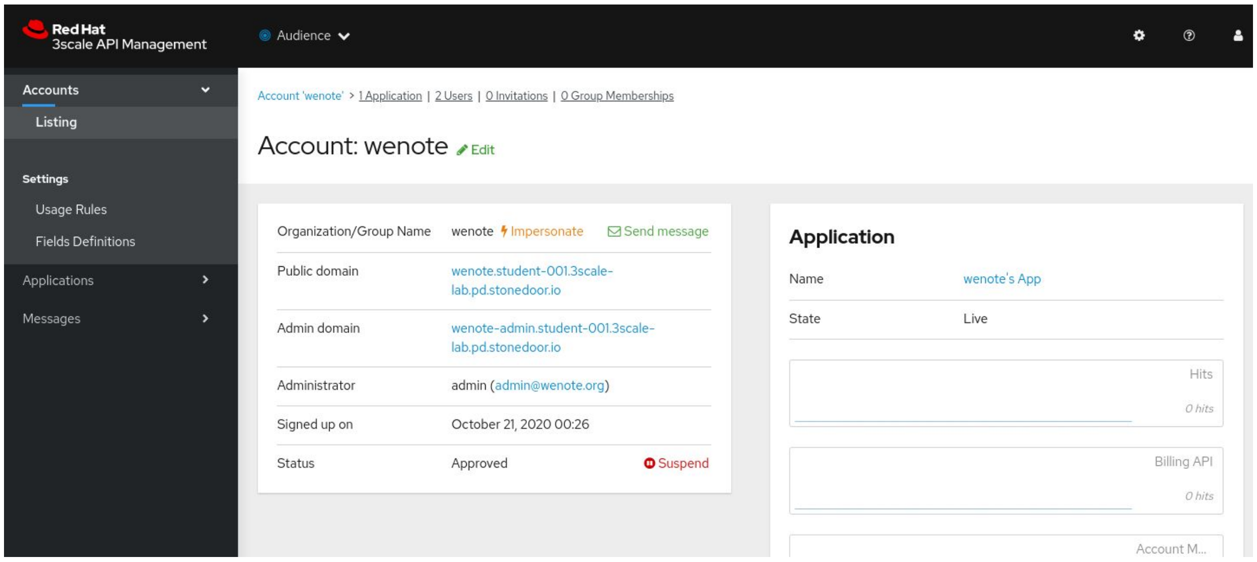
5. Activate the account by **clicking** back to the "Accounts->Listing" tab and then **clicking** the activate button to the far right of the account name in the listing:



Accounts

	Group/Org.	Admin	Signup Date	State
	wenote	admin	21 Oct, 2020	Approved

6. **Click** on the activated account and note that both a public and admin domain were created within the base (wildcard) domain you specified when installing 3Scale:



Account: wenote

Organization/Group Name	wenote
Public domain	wenote.student-001.3scale-lab.pd.stonedoor.io
Admin domain	wenote-admin.student-001.3scale-lab.pd.stonedoor.io
Administrator	admin (admin@wenote.org)
Signed up on	October 21, 2020 00:26
Status	Approved

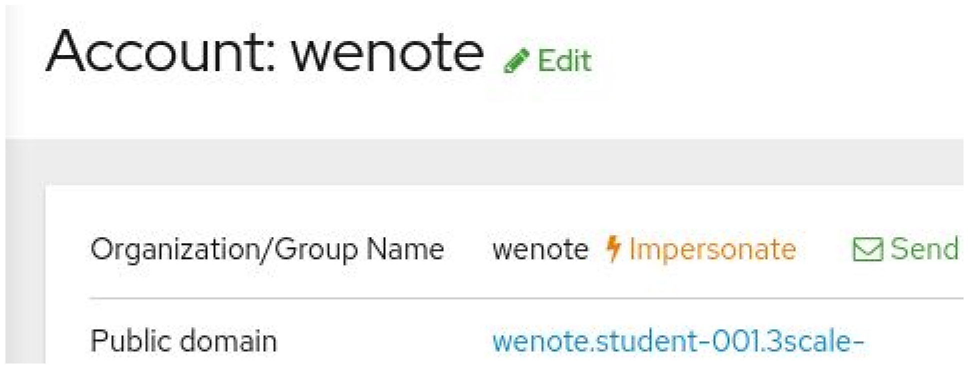
Application

Name	wenote's App
State	Live

3.2 Create a New Product

In order to grant access to external developers, the newly created administrator must create a Product and the corresponding objects in 3Scale. The following steps demonstrate how to setup WeNote Product in 3Scale.

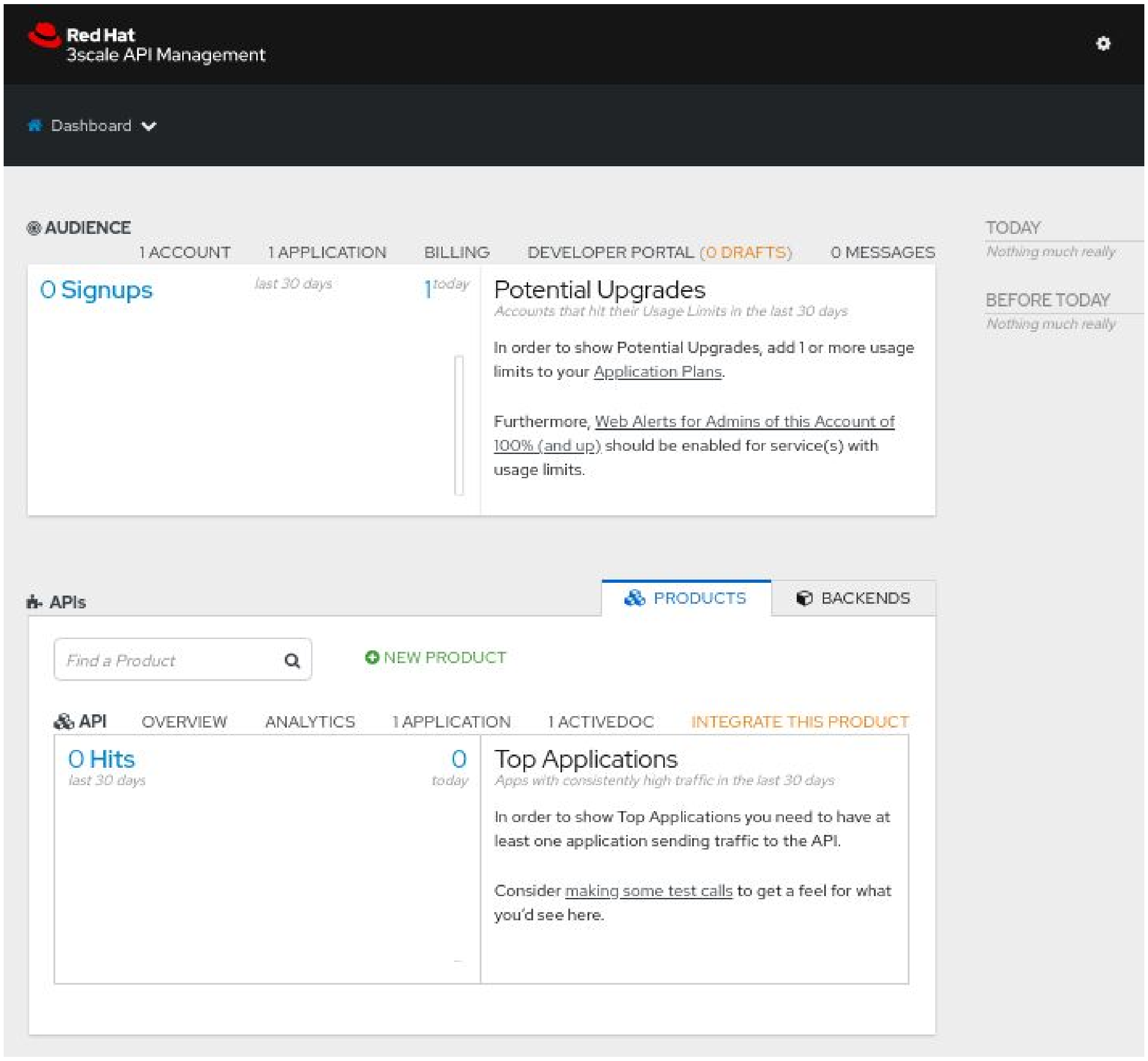
1. **Click** on the orange Impersonate link. This will take you to the admin console for the WeNote account and mimic the Admin for a development group:


Account: wenote

Organization/Group Name	wenote
Public domain	wenote.student-001.3scale-

A new browser tab should open, already logged in, to the admin dashboard for the WeNote account. **Close** the other tab that was connected to the master dashboard.

2. The dashboard lists important information such as the number of sign-ups and baseline API hits. This will be used by the Admin to track usage and throughput of the various published endpoints:



Potential Upgrades

Accounts that hit their Usage Limits in the last 30 days

In order to show Potential Upgrades, add 1 or more usage limits to your Application Plans.

Furthermore, Web Alerts for Admins of this Account of 100% (and up) should be enabled for service(s) with usage limits.

Top Applications

Apps with consistently high traffic in the last 30 days

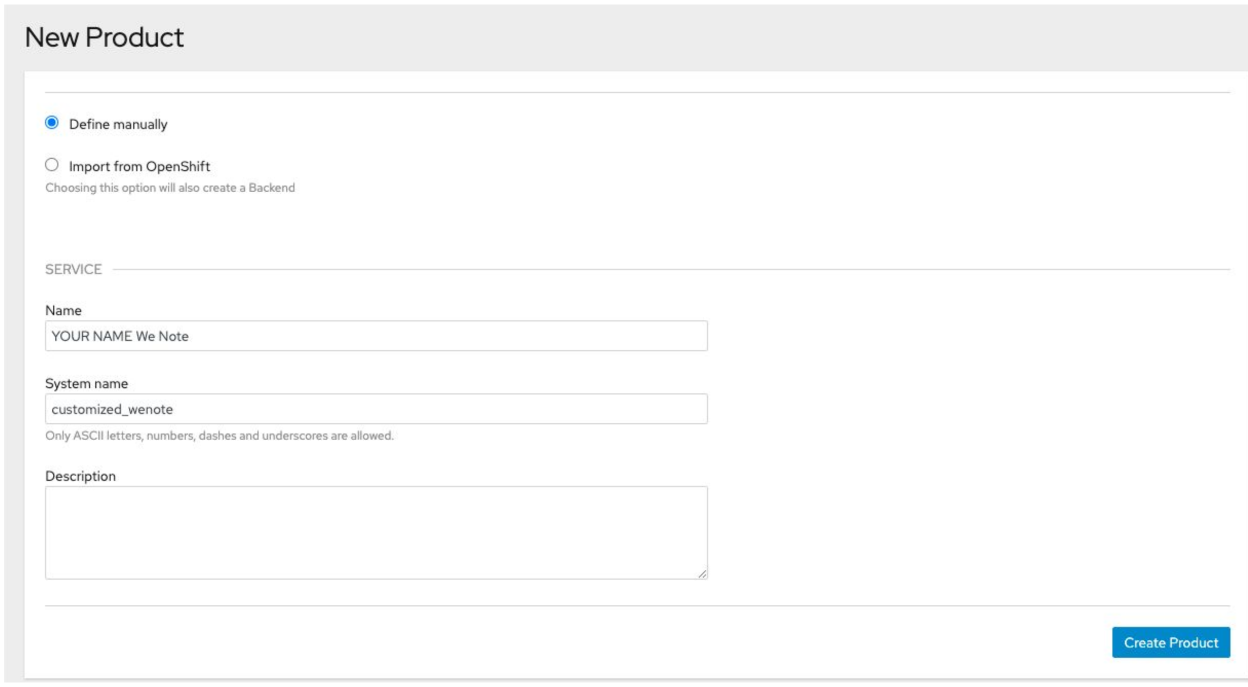
In order to show Top Applications you need to have at least one application sending traffic to the API.

Consider making some test calls to get a feel for what you'd see here.

3. Following account creation, 3Scale creates a default application and subscribed developer. We however want to create a new product from the beginning. From the WeNote admin console (not the master console), locate and click the NEW PRODUCT button.

4. For this exercise we will define manually the product - **complete** the form using your name where indicated and **click** the Create Product button
   - Name: ***<Your_Name>*** **`We Note`**
   - System name: ***<firstname-lastname>*** **`_wenote`**

- Description: `(optional field)`



New Product

● Define manually

○ Import from OpenShift
Choosing this option will also create a Backend

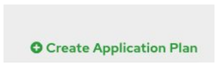
SERVICE

Name: YOUR NAME We Note

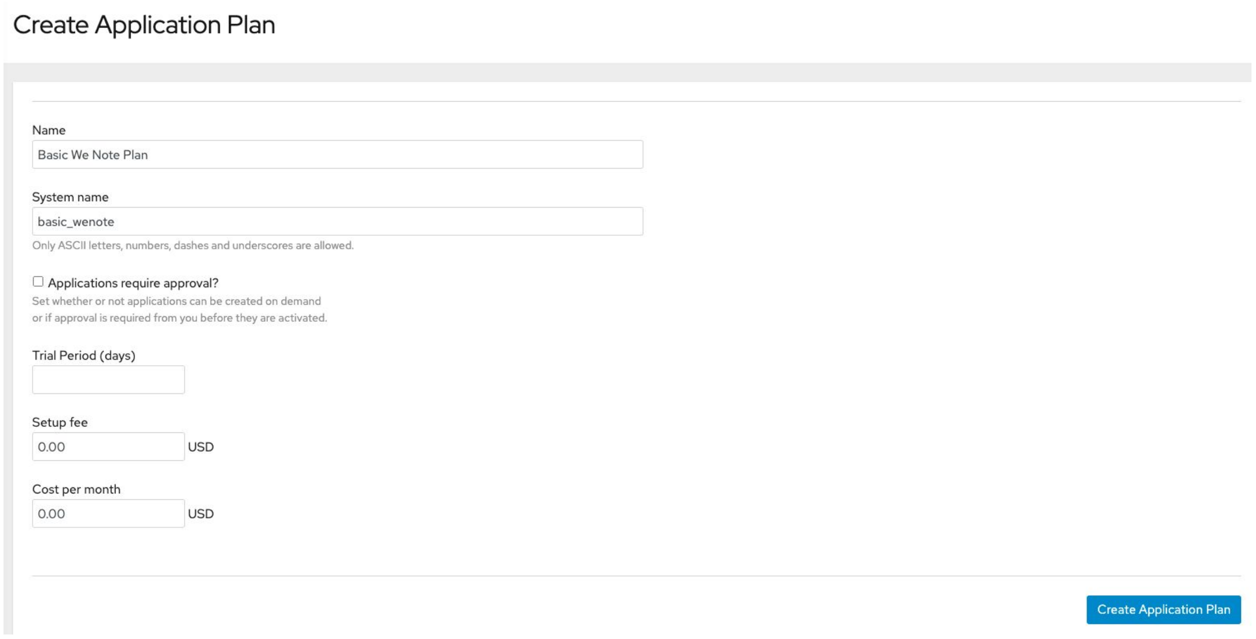
System name: customized_wenote
Only ASCII letters, numbers, dashes and underscores are allowed.

Description:

5. The Product Overview screen is presented.
   Click on the Create Application Plan button to get started:



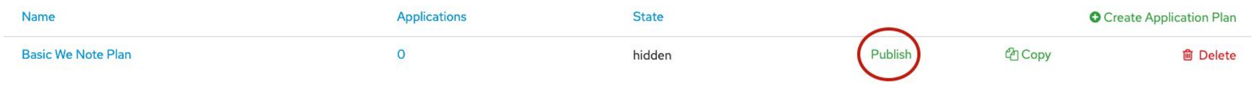
6. These plans are for the subscription to the application. **Complete** the form using the following values::
   - Name: **`Basic We Note plan`**
   - System name: **`basic_wenote`**
   - Applications require approval?: ***`<unchecked>`***
   - Trial Period: ***`<blank>`***
   - Setup fee: ***`<blank>`***
   - Cost per month: ***`<blank>`***



Create Application Plan

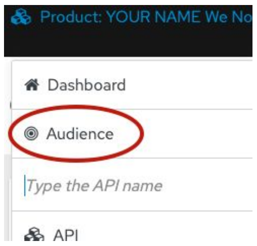
Name	Basic We Note Plan
System name	basic_wenote
	Only ASCII letters, numbers, dashes and underscores are allowed.
☐ Applications require approval?	Set whether or not applications can be created on demand or if approval is required from you before they are activated.
Trial Period (days)	
Setup fee	0.00 USD
Cost per month	0.00 USD

7. You will now see the Application Plans Screen and should see your newly created plan listed. **Click** the Publish button to make your plan active:

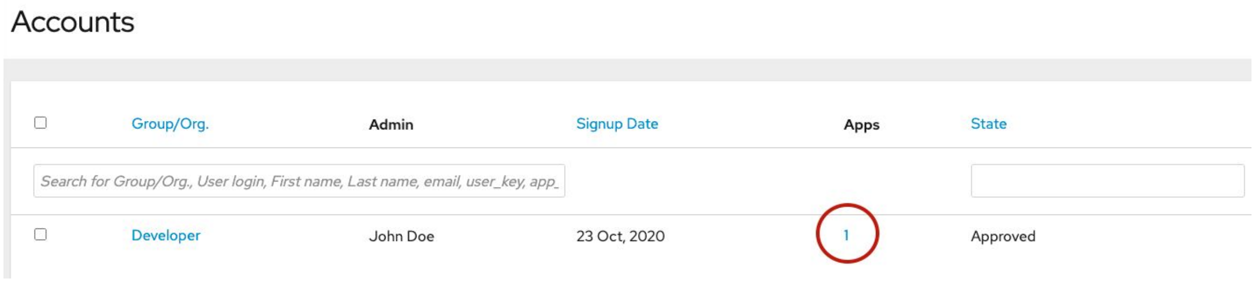


Name	Applications	State			
Basic We Note Plan	0	hidden	Publish	Copy	Delete

8. Create an application definition to use this plan. From the top drop down menu **select** Audience and the list of account will be shown (only the auto-created Developer account exists so far):



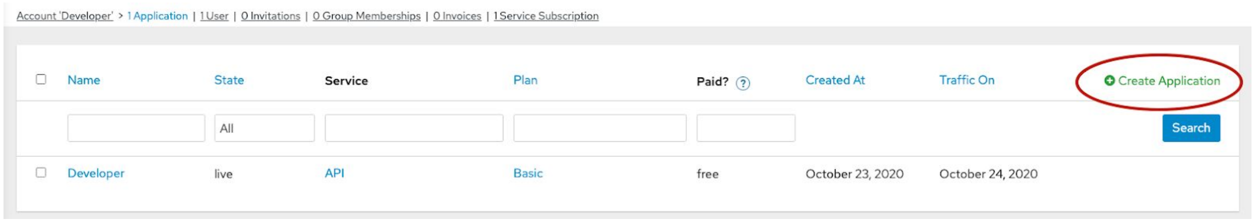
9. Click on the (**1**) under the Apps column:



Accounts

☐	Group/Org.	Admin	Signup Date	Apps	State
	Search for Group/Org., User login, First name, Last name, email, user_key, app_				
☐	Developer	John Doe	23 Oct, 2020	1	Approved

10. The screen displays the Applications this Developer is subscribed to. You can see the default account named Developer is live and has a basic plan. **Click** on the Create Application button to create your application:

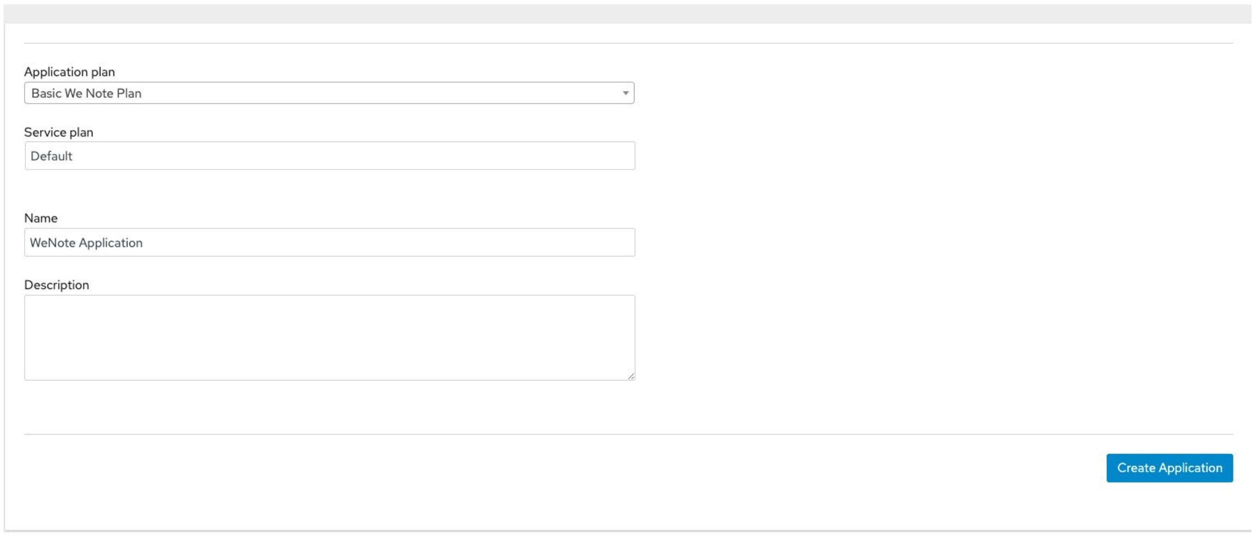


Account 'Developer' > 1 Application | 1 User | 0 Invitations | 0 Group Memberships | 0 Invoices | 1 Service Subscription

	Name	State	Service	Plan	Paid?	Created At	Traffic On
		All					
☐	Developer	live	API	Basic	free	October 23, 2020	October 24, 2020

11. **Complete** out the form and **click** the Create Application button:
    - Application Plan(select): **`Basic We Note Plan`**
    - Service Plan: **`Default`**
    - Name: **`WeNote Application`**
    - Description: ***`<optional>`***

New Application



Application plan: Basic We Note Plan

Service plan: Default

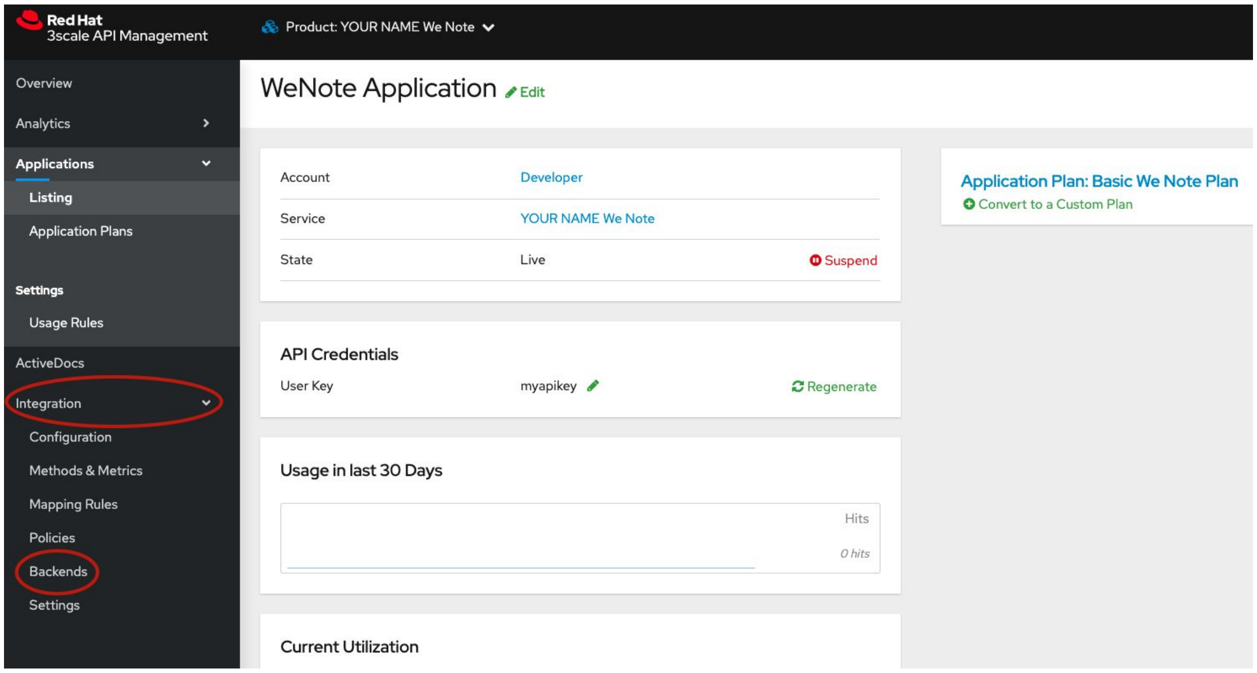
Name: WeNote Application

Description:

The WeNote Application Listing is displayed.

12. Note that the API Credentials box shows a generated User Key. This value will be needed later for testing. **Copy** and **save** the provided value, or optionally **click** the edit icon and enter a valid, memorable key such as: `myapikey`

13. Now that the Administrator has created a Product, Application and Plan, the administrator needs to define how this Application will call the WeNote API created in the Openshift cluster. On the left hand side, **select**: Integration-> Backends:



WeNote Application

Account	Developer	
Service	YOUR NAME We Note	
State	Live	Suspend

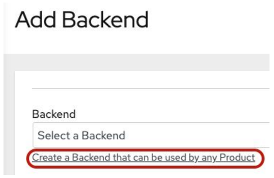
API Credentials

User Key	myapikey	Regenerate

Usage in last 30 Days

Current Utilization

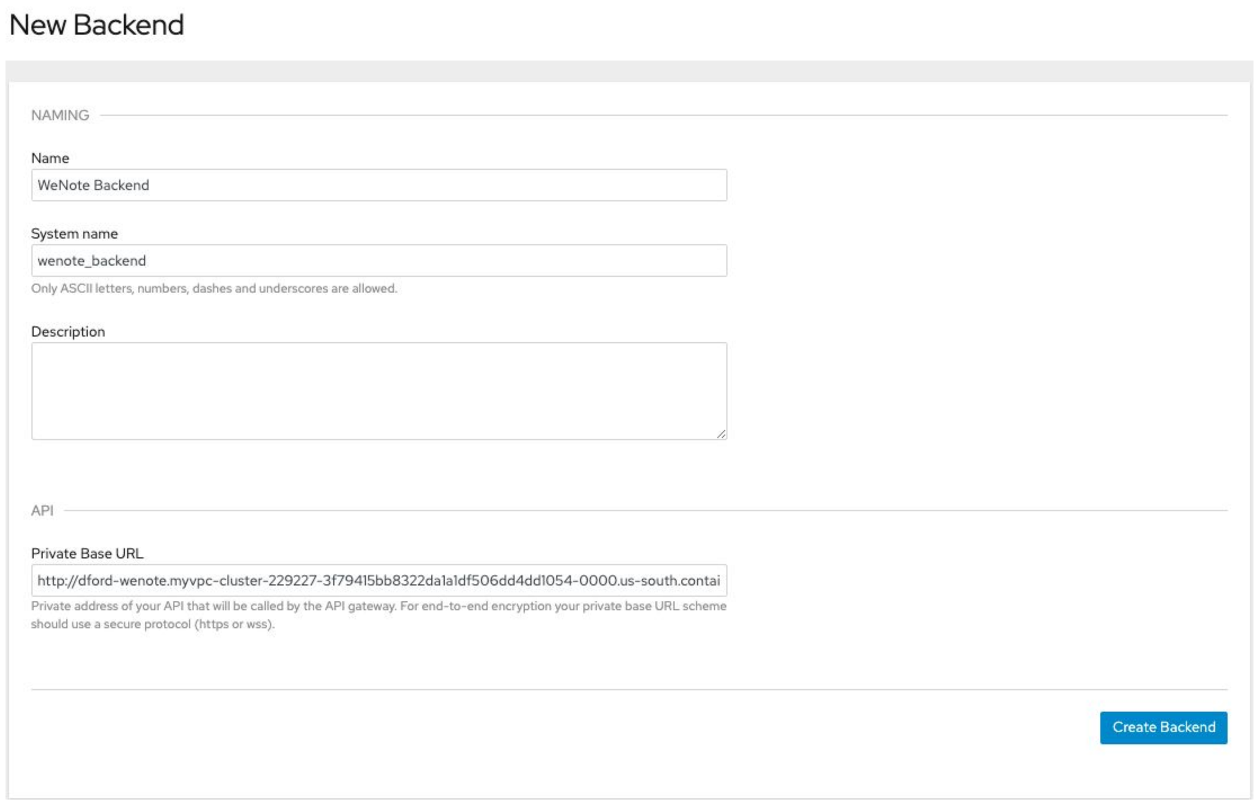
14. Create a new backend by **clicking** on “Add Backend”. Creation requires selecting a backend and providing a path. Since only the default API Backend currently exists, **click** the “Create a Backend that can be used by any Product” link (text) under the Backend selection box to create a new one.



Fill the “New Backend” form as follows and then **click** create:

- Name: **`WeNote Backend`**
- System name: **`wenote_backend`**
- Description: ***`<optional>`***
- Private Base URL:
`http://`*`<firstname-lastname>`*`-wenote.myvpc-cluster-229227-3f79415bb8322da1a1df506dd4dd1054-0000.us-south.containers.appdomain.cloud`

This URL is the same that was created in the `demogateway.yaml` previously and the one that was used in the curl command (just the base URL, NOT the path):


New Backend

NAMING

Name
WeNote Backend

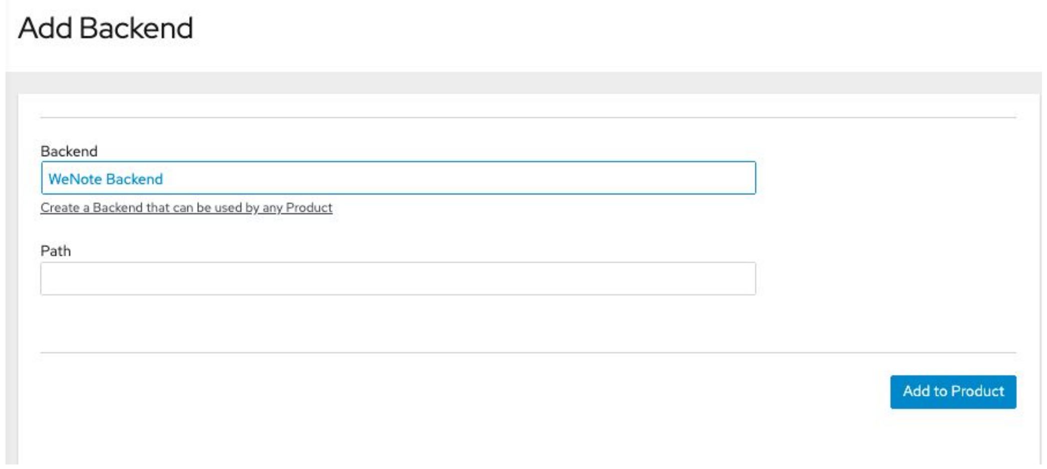
System name
wenote_backend
Only ASCII letters, numbers, dashes and underscores are allowed.

Description

API

Private Base URL
http://dford-wenote.myvpc-cluster-229227-3f79415bb8322da1a1df506dd4dd1054-0000.us-south.contai
Private address of your API that will be called by the API gateway. For end-to-end encryption your private base URL scheme should use a secure protocol (https or wss).

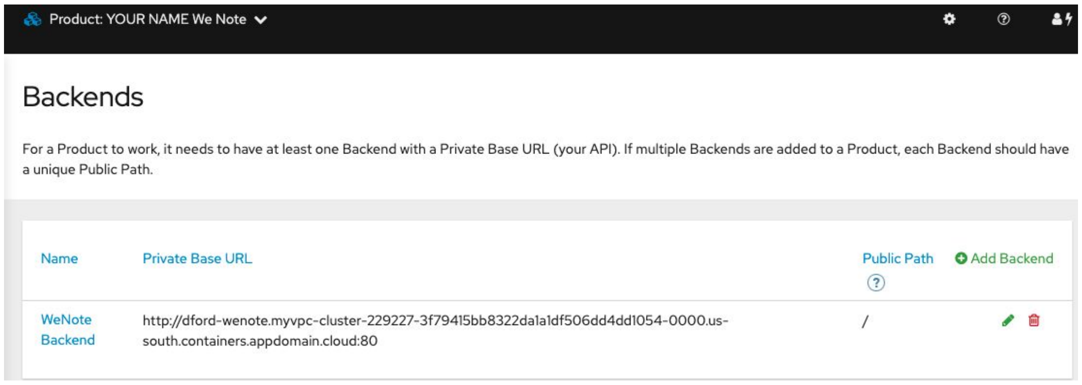
15. Now that the WeNote backend is created we must associate it with the main API. From the top dropdown menu, again **select** the WeNote Product (created at the beginning of this section) and then navigate on the left side menu to Integration->Backends. **Click** the Add Backend (Add Backend) button and in the Backend field, **select** the newly created WeNote backend. Leave the Path blank and **click** the Add to Product button.


Add Backend

Backend
WeNote Backend
Create a Backend that can be used by any Product

Path

16. You should now see the backend under your WeNote Product:



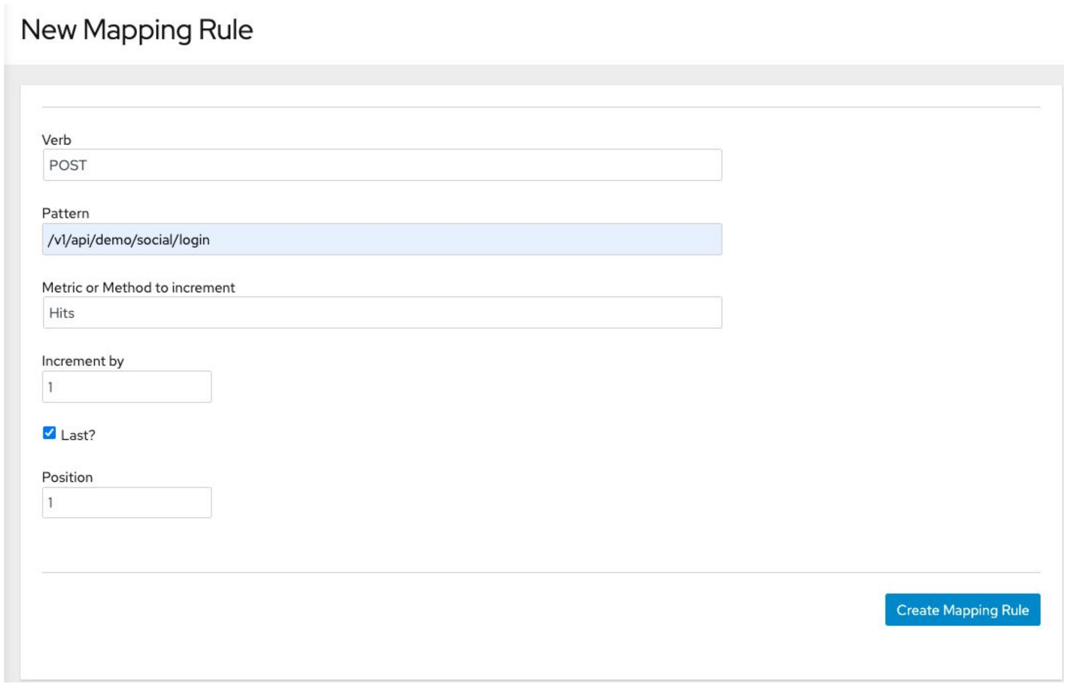
Backends

For a Product to work, it needs to have at least one Backend with a Private Base URL (your API). If multiple Backends are added to a Product, each Backend should have a unique Public Path.

Name	Private Base URL	Public Path	Add Backend
WeNote Backend	http://dford-wenote.myvpc-cluster-229227-3f79415bb8322da1a1df506dd4dd1054-0000.us-south.containers.appdomain.cloud:80	/	

17. In order to track each API individually, you must setup a Mapping for each API. This will allow the administrator to see metrics for each API as opposed to all APIs together. The administrator can also use methods and various Metrics to get really granular. For the WeNote application we will keep it simple and create a mapping for the default metric of "Hits". **Select** your WeNote Product from the top drop down menu, from the left **navigate** to Integration -> Mapping Rules. **Click** the Add Mapping Rule button:

18. Complete the form as follows and click Create Mapping Rule button
    - Verb: POST
    - Pattern: /v1/api/demo/social/login
    - Metric or Method to increment: Hits
    - Incremented by: 1
    - Last: <*selected*>
    - Position: 1


New Mapping Rule

Verb: POST

Pattern: /v1/api/demo/social/login

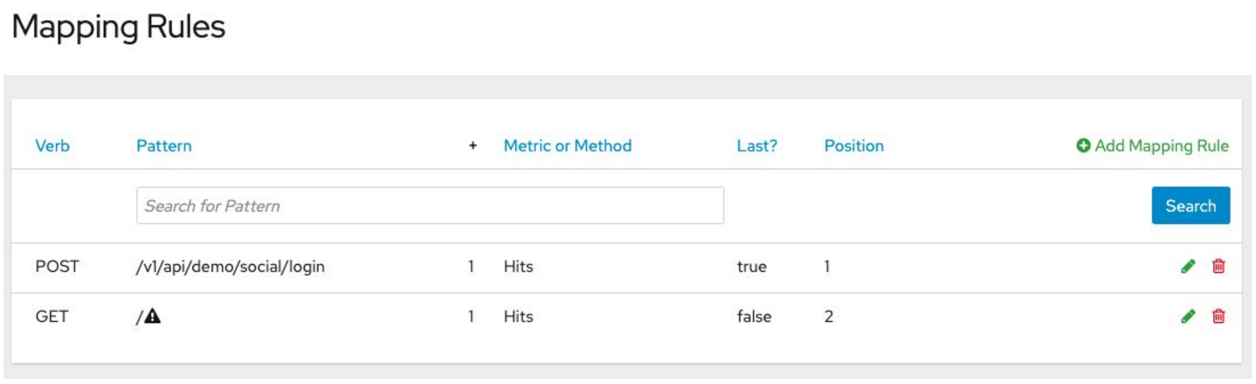
Metric or Method to increment: Hits

Increment by: 1

Last? ✓

Position: 1

You will see the Mapping Rules displayed:

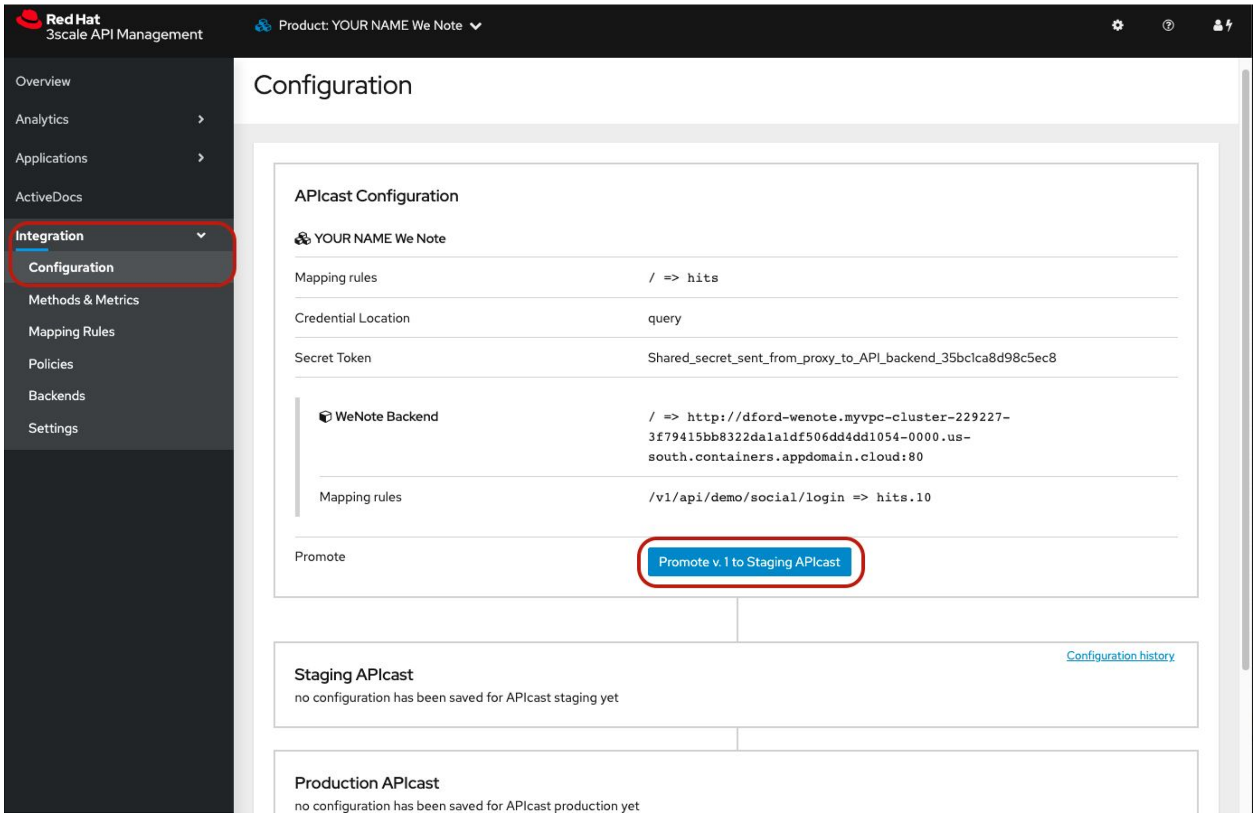

Mapping Rules

Verb	Pattern	+	Metric or Method	Last?	Position
POST	/v1/api/demo/social/login	1	Hits	true	1
GET	/⚠	1	Hits	false	2

3.3 Promoting to Staging, Production and Testing the WeNote API

1. The product is now fully defined. However, it is not available for external consumption (published). 3Scale follows a proper dev -> staging -> prod workflow, where APIs are promoted through stages.
   As the administrator, promote the WeNote API from the 3Scale, **select** from the sidebar, Integration -> Configuration. From the APICast Configuration box, **click** on the blue

"Promote the v.*x* to Staging API" button:



Configuration

APIcast Configuration

YOUR NAME We Note

Mapping rules	`/ => hits`
Credential Location	query
Secret Token	Shared_secret_sent_from_proxy_to_API_backend_35bc1ca8d98c5ec8
WeNote Backend	`/ => http://dford-wenote.myvpc-cluster-229227-3f79415bb8322da1a1df506dd4dd1054-0000.us-south.containers.appdomain.cloud:80`
Mapping rules	`/v1/api/demo/social/login => hits.10`
Promote	Promote v. 1 to Staging APIcast

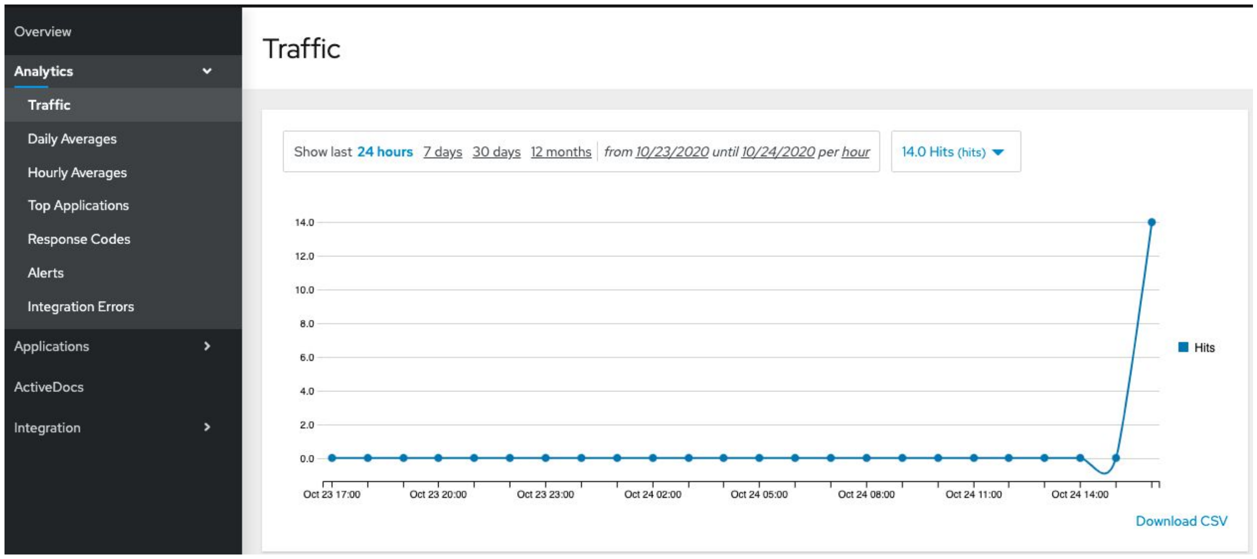
Staging APIcast
no configuration has been saved for APIcast staging yet

Production APIcast
no configuration has been saved for APIcast production yet

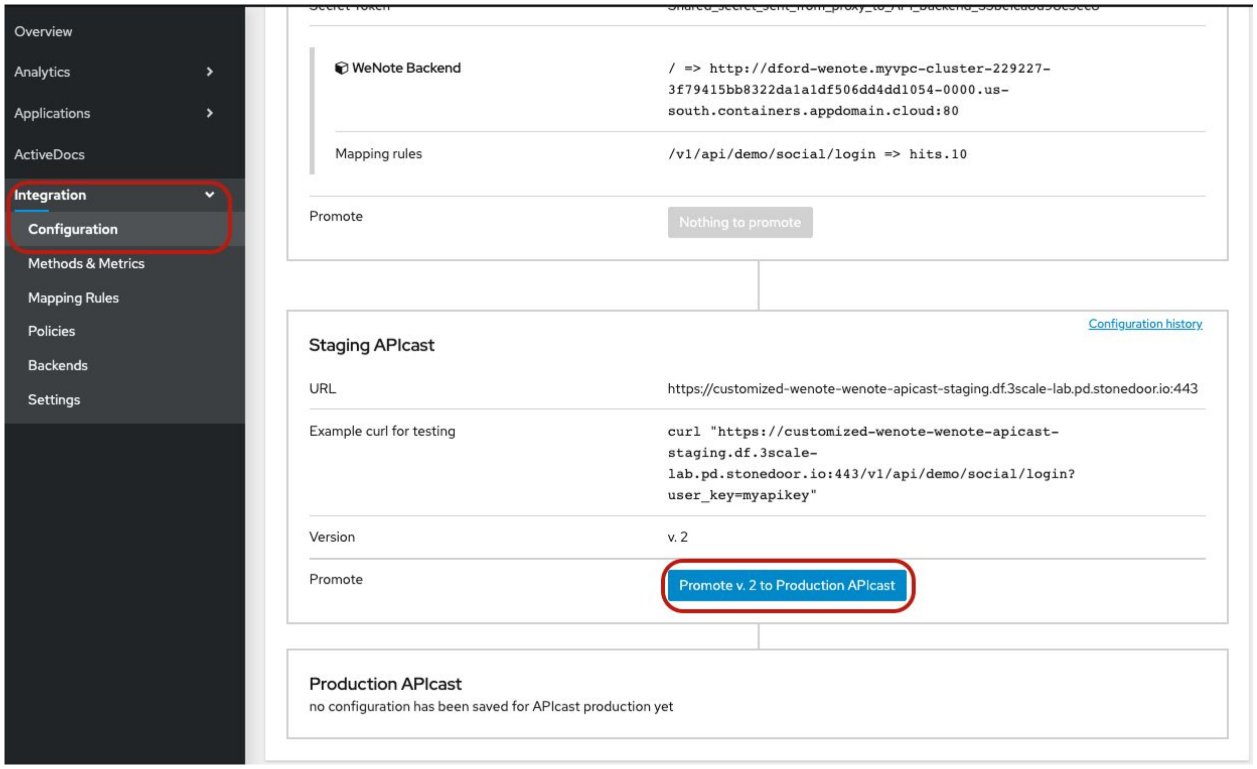
2. To test the API, use the URL for accessing the API shown under the "Example curl for testing" section in APIcast Configuration (note that your API key is already being passed as the user_keyparameter). **Cut** and **paste** the full command, then **run** it in your Cloud Shell terminal, **adding** the option to accept unknown TLS certificates; for example:

```
$ curl -k -X POST
https://<firstname-lastname>-wenote-wenote-apicast-staging.df.3scale-lab.pd.st
onedoor.io:443/v1/api/demo/social/login?user_key=myapikey -d '{}' -H
'Content-Type: application/json'
{"token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJleHAiOjE2MDM1NTc4MjV9.NtC24oO
8vCGPQaf_TaUdFWvIq1ohwG8du-_xA-MgVwLnA1UtolzD7Kqk9Y3rb2L4imFy8Z-JQoJu11XXzfvrRw
","enabled":[]}
```

3. From the sidebar menu navigate and to the Analytics -> Traffic you will see that the system has captured the hits from the curl(s) you performed:



4. After having successful tests in staging, now **navigate** on the sidebar to Integration-> Configuration and **click** the Promote to Production v.X to Production APIcast:



WeNote Backend	/ => http://dford-wenote.myvpc-cluster-229227-3f79415bb8322da1a1df506dd4dd1054-0000.us-south.containers.appdomain.cloud:80
Mapping rules	/v1/api/demo/social/login => hits.10

Staging APIcast

URL	https://customized-wenote-wenote-apicast-staging.df.3scale-lab.pd.stonedoor.io:443
Example curl for testing	curl "https://customized-wenote-wenote-apicast-staging.df.3scale-lab.pd.stonedoor.io:443/v1/api/demo/social/login?user_key=myapikey"
Version	v. 2

Production APIcast
no configuration has been saved for APIcast production yet

5. Now you can go back to the Kiali dashboard and see additional traffic.

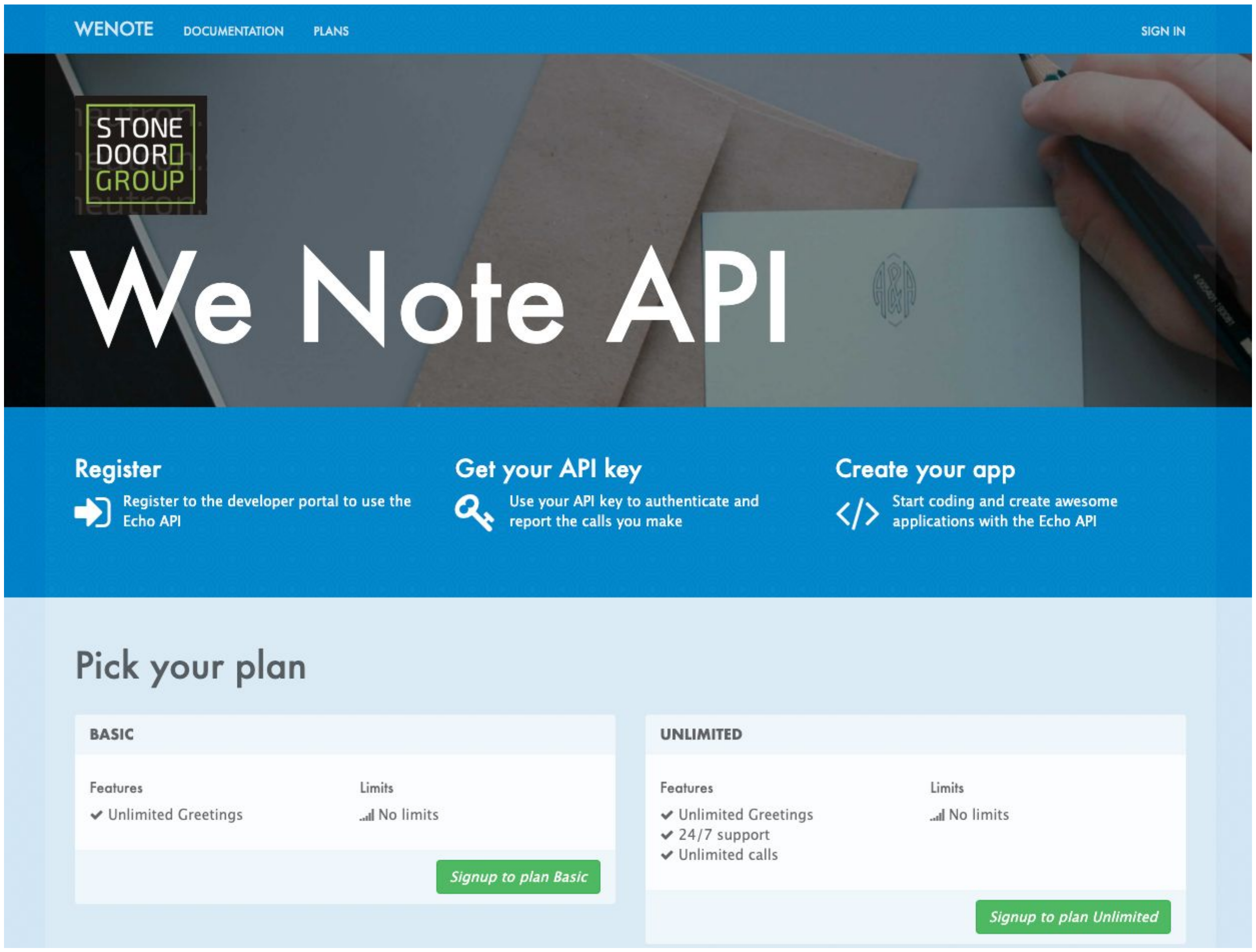
4.0 Conclusion

Now that we have installed 3Scale API, connected WeNote to Service Mesh and 3Scale and tested our application API, now a third party can request access to WeNote and use the endpoint.

Appendix A - Example External Application Portal

The 3Scale application has an outside facing portal where 3rd parties who want to utilize the published API's can register and get access to them. The Administrator can define documentation using Liquid and expose a portal for testing the APIs using Swagger:

Here is an example of a Landing Page for the portal:



Register
Register to the developer portal to use the Echo API

Get your API key
Use your API key to authenticate and report the calls you make

Create your app
Start coding and create awesome applications with the Echo API

Pick your plan

BASIC

Features
✓ Unlimited Greetings

Limits
No limits

Signup to plan Basic

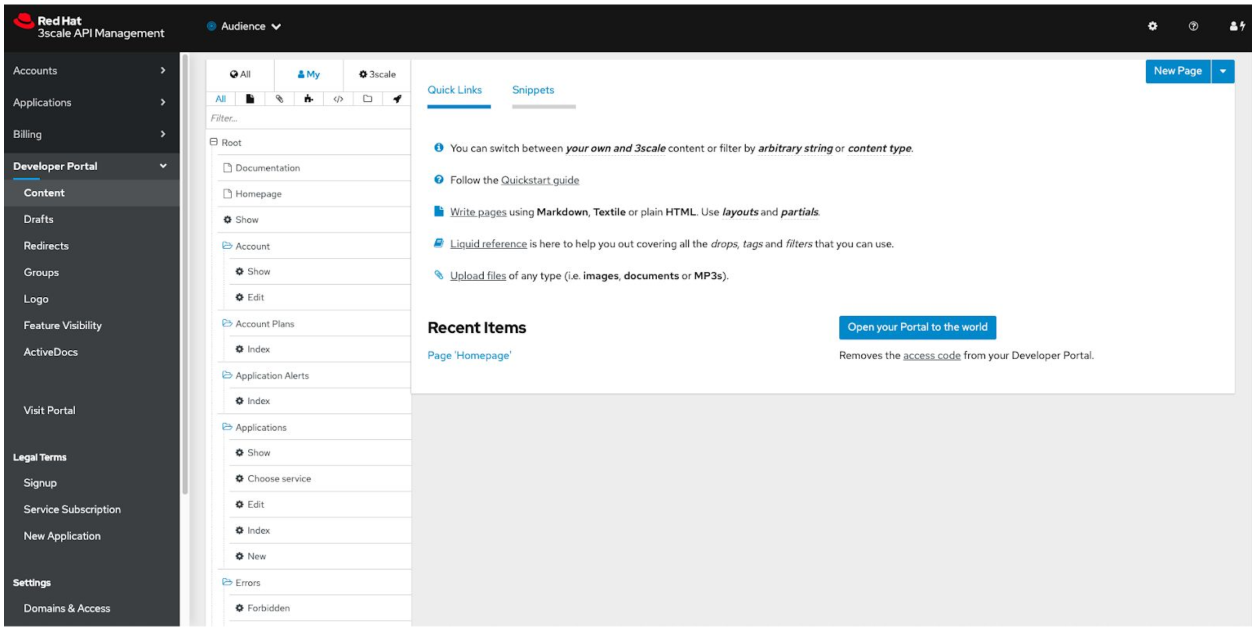
UNLIMITED

Features
✓ Unlimited Greetings
✓ 24/7 support
✓ Unlimited calls

Limits
No limits

Signup to plan Unlimited

This portal is maintained by the Administrator and has quite extensive capabilities:



You can switch between ***your own and 3scale*** content or filter by ***arbitrary string*** or ***content type***.

Follow the Quickstart guide

Write pages using **Markdown**, **Textile** or plain **HTML**. Use ***layouts*** and ***partials***.

Liquid reference is here to help you out covering all the *drops*, *tags* and *filters* that you can use.

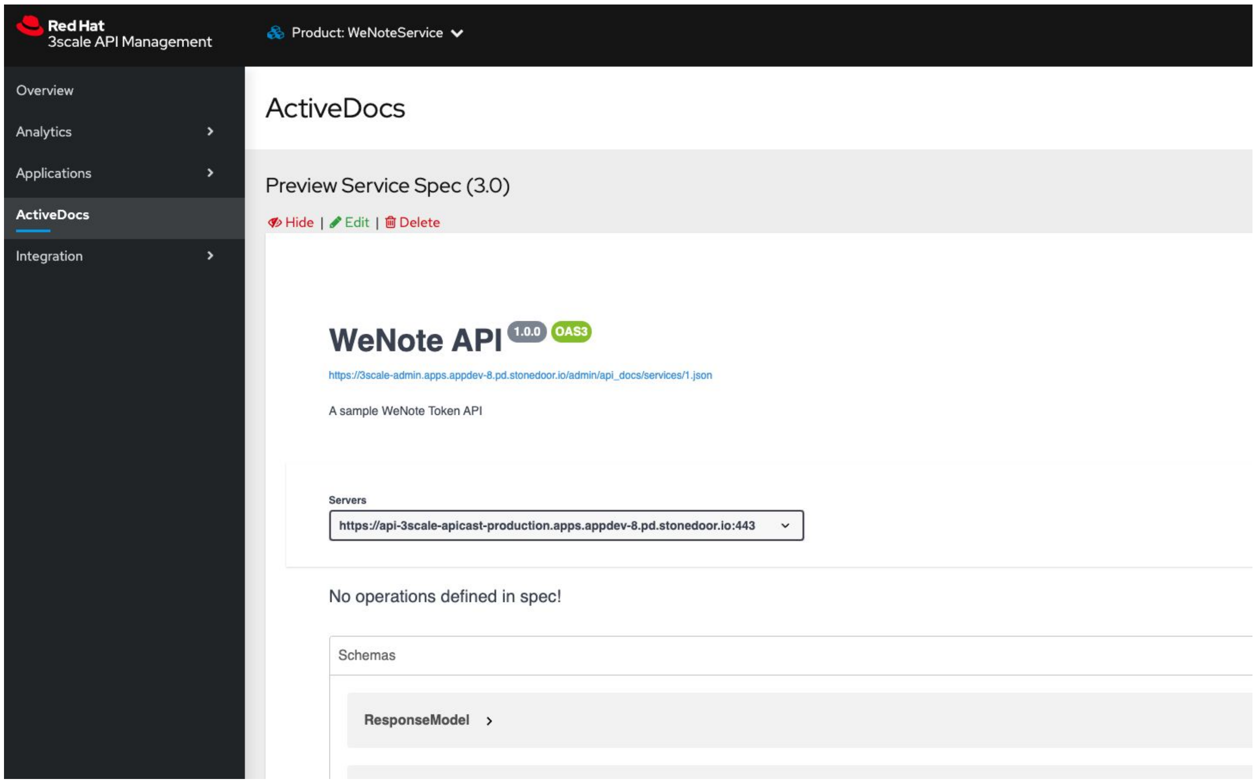
Upload files of any type (i.e. **images**, **documents** or **MP3s**).

Recent Items

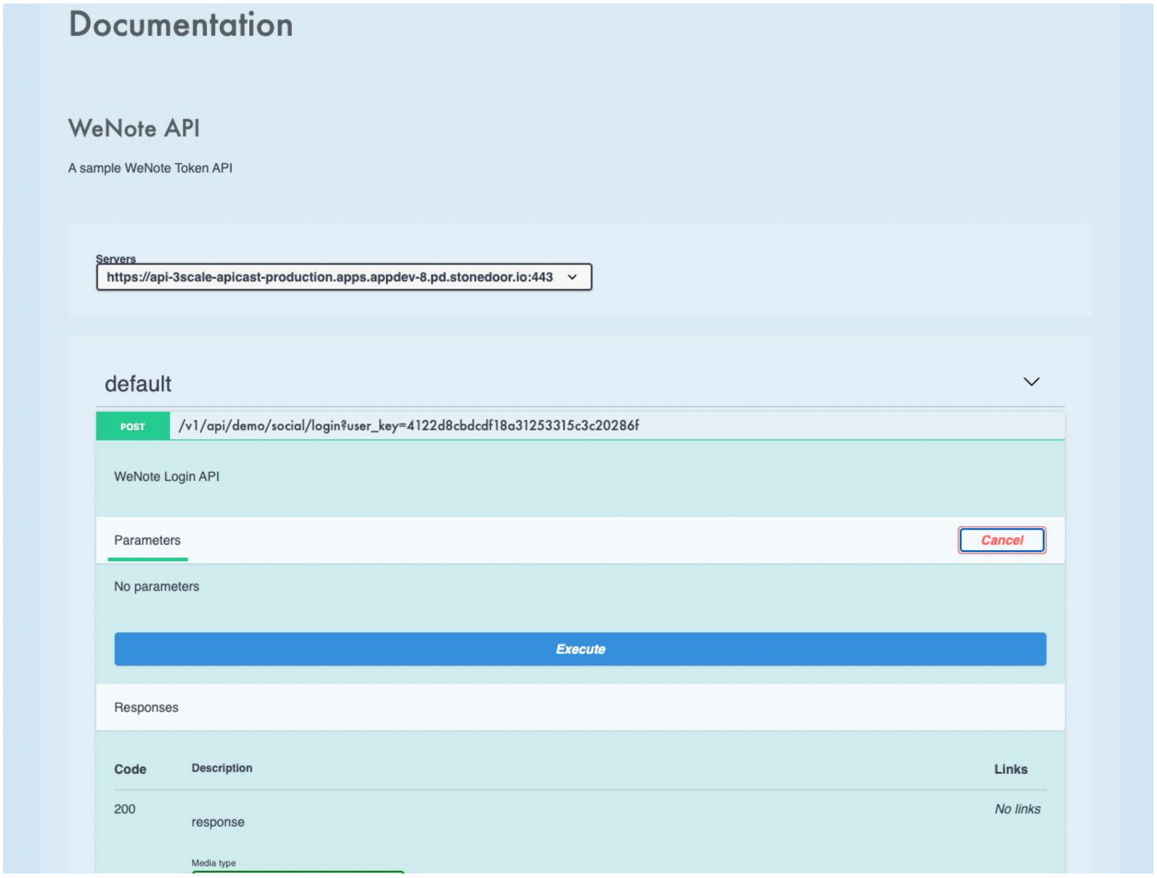
Page 'Homepage'

Removes the access code from your Developer Portal.

The Administrator can edit, preview and publish the swagger docs:


Red Hat
3scale API Management
Product: WeNoteService
Overview
Analytics
Applications
ActiveDocs
Integration
ActiveDocs
Preview Service Spec (3.0)
Hide | Edit | Delete
WeNote API
1.0.0
OAS3
https://3scale-admin.apps.appdev-8.pd.stonedoor.io/admin/api_docs/services/1.json
A sample WeNote Token API
Servers
https://api-3scale-apicast-production.apps.appdev-8.pd.stonedoor.io:443
No operations defined in spec!
Schemas
ResponseModel

The 3rd party developer can test the APIs:



Documentation

WeNote API

A sample WeNote Token API

Servers
https://api-3scale-apicast-production.apps.appdev-8.pd.stonedoor.io:443

default

POST /v1/api/demo/social/login?user_key=4122d8cbdcdf18a31253315c3c20286f

WeNote Login API

Parameters

No parameters

Code	Description	Links
200	response	*No links*

Media type

Appendix B - Additional Documentation/Information

Red Hat overview of 3Scale
https://www.redhat.com/en/technologies/jboss-middleware/3scale

Red Hat 3Scale documentation
https://access.redhat.com/documentation/en-us/red_hat_3scale_api_management/

Training & certification

Build and Administer APIs with Red Hat 3scale API Management with exam (AD241)

Red Hat Certified Specialist in Enterprise Application Server Administration